Technical Report No. 148
Copy 17 of 35

VOLUME II

SILVER SPRING, MARYLAND
SANTA MONICA, CALIFORNIA
ATLANTA, GEORGIA

PROJECT PHAROS
ANNUAL COMPREHENSIVE REPORT ON SENSITIVITY ANALYSIS OF OPERATIONAL EFFECTIVENESS: CW/BW WEAPONS SYSTEMS

by

Roger C. Eyler, Reynold Greenstone,
Thomas L. Newberry, and Cephus T. Patch

31 August 1961

Prepared for Director, U.S. Army Chemical Corps
Operations Research Group Army Chemical Center, Maryland
under Contract No. DA 18-108-CML-6554
Order No. CPI-902

OPERATIONS RESEARCH, *Incorporated*

SILVER SPRING, MARYLAND

PROJECT PHAROS
ANNUAL COMPREHENSIVE REPORT ON SENSITIVITY ANALYSIS OF OPERATIONAL EFFECTIVENESS: CW/BW WEAPONS SYSTEMS

by

Roger C. Eyler, Reynold Greenstone,
Thomas L. Newberry, and Cephas T. Patch

31 August 1961

Prepared for Director, U.S. Army Chemical Corps
Operations Research Group Army Chemical Center, Maryland
under Contract No. DA 18-108-CML-6554
Order No. CPI-902

TABLE OF CONTENTS

LIST OF FIGURES

I. INTRODUCTION

1.1 This volume of the comprehensive report presents the specific computer simulation that has been devised and tested during the contract year.

1.2 In Section II the conventional computer flow charts are presented. In Section III the symbols used in the flow charts are defined and a description of the flow charts is given. This description takes the reader step-by-step through the simulation explaining the purpose of each computer operation.

1.3 In Section IV the literal computer program for the simulation is presented. In order to achieve generality in application, two versions of the program, one in ALGOL and the other in FORTRAN, have been prepared and are presented one after the other.

II. FLOW CHARTS FOR COMPUTER SIMULATION OF ARTILLERY-FIRED GB SHELLS

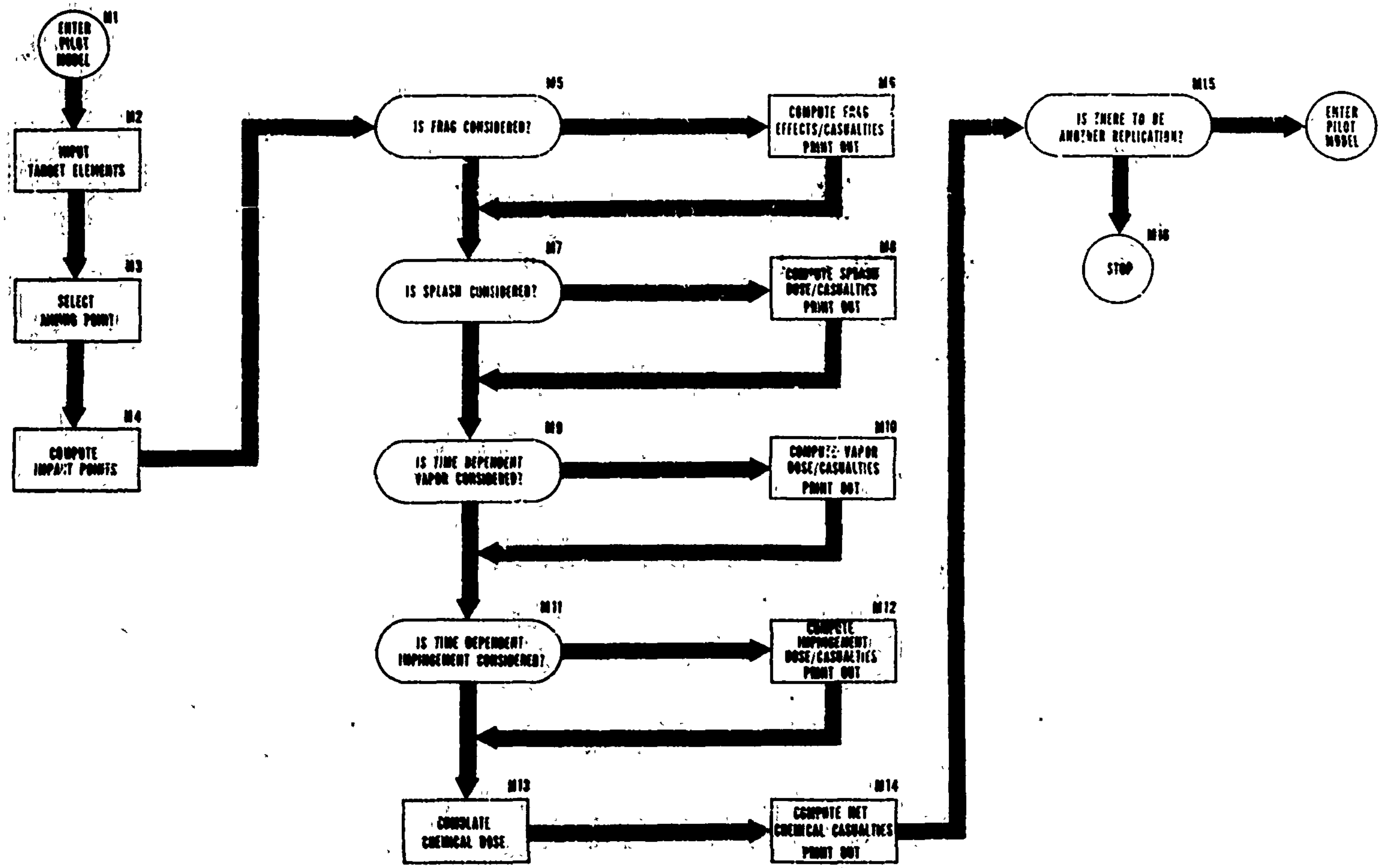
2.1 This section presents the seven flow charts which provide a conventional representation of the computer simulation.

2.2 With the advent of the new computer languages such as ALGOL and FORTRAN, certain simplifications have become available in computer programing which tend to cause minor differences between the flow charts and the programs based on them. For example, the FOR instruction in ALGOL supplants the conventional loop program illustrated in the flow charts. The instruction: FOR N = (1, 1, 12); BEGIN....takes the place of the loop: SET N = 1; PERFORM OPERATION ON VARIABLE$_N$; Is N = 12?; IF N = 12, EXIT; IF N $\neq$ 12, ADD 1 TO N, PERFORM OPERATION ON VARIABLE$_N$....

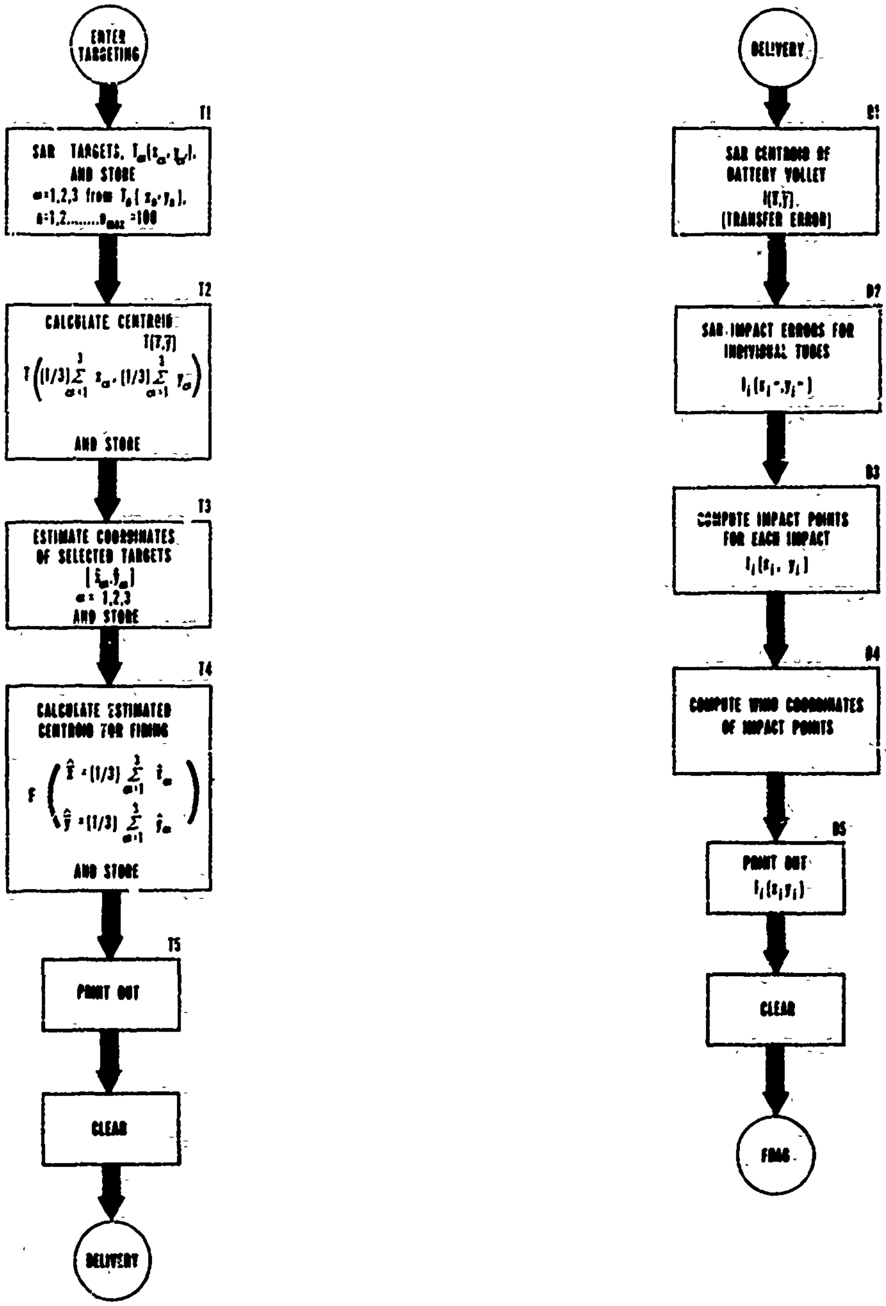
2.3 When several loops are contained one inside the other, the flow charts tend to clarify the order of operations which can become obscure when embodied in the computer programs.

2.4 Flow Chart 1 gives an over-all view of the simulation. The succeeding charts present: (a) targeting and delivery of the GB shells, (b) determination of the fragmentation effect, (c) determination of the splash effect, (d) determination of the GB vapor effect. (This last appears on three charts. On the first chart of the series a geometric determination of whether a target is in the path of the GB vapor cloud is made. On the second chart the determination of breathing parameters, response delay time, breathing phase at impact time, masking time, breathhold time, and gasp time is made. On the third chart the inhaled vapor dose is calculated.)

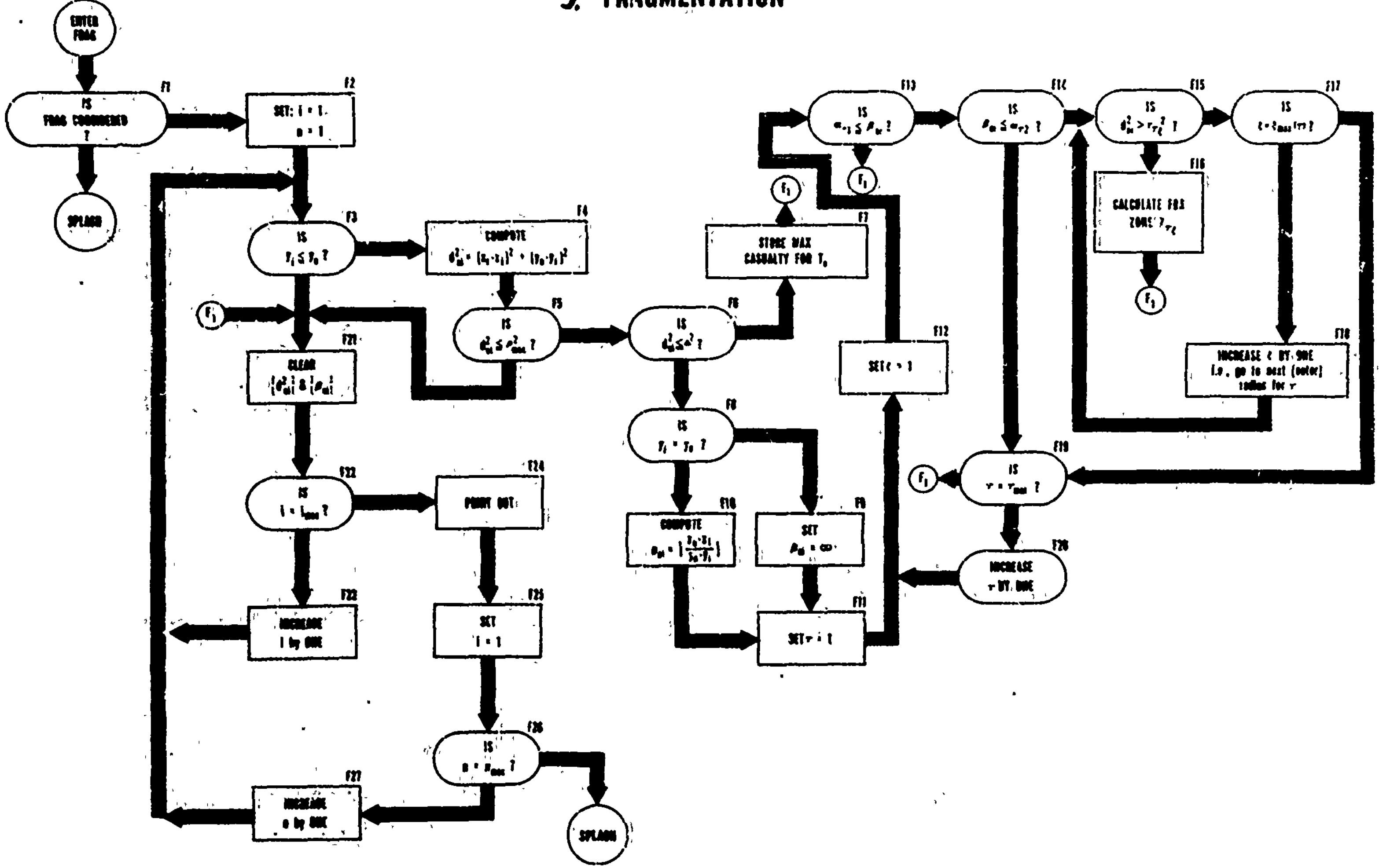
1. MASTER FLOW CHART



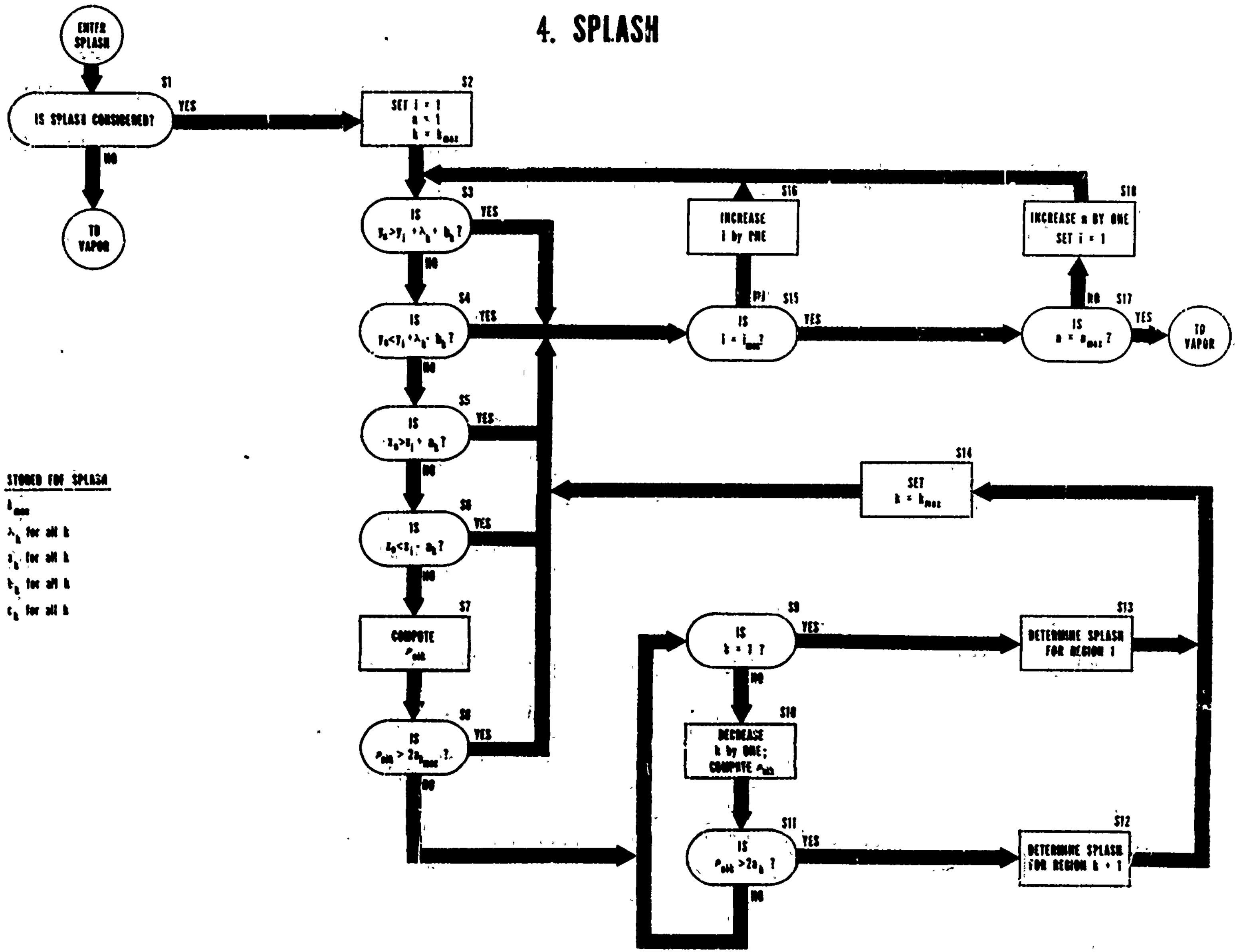
2. TARGETING AND DELIVERY



3. FRAGMENTATION



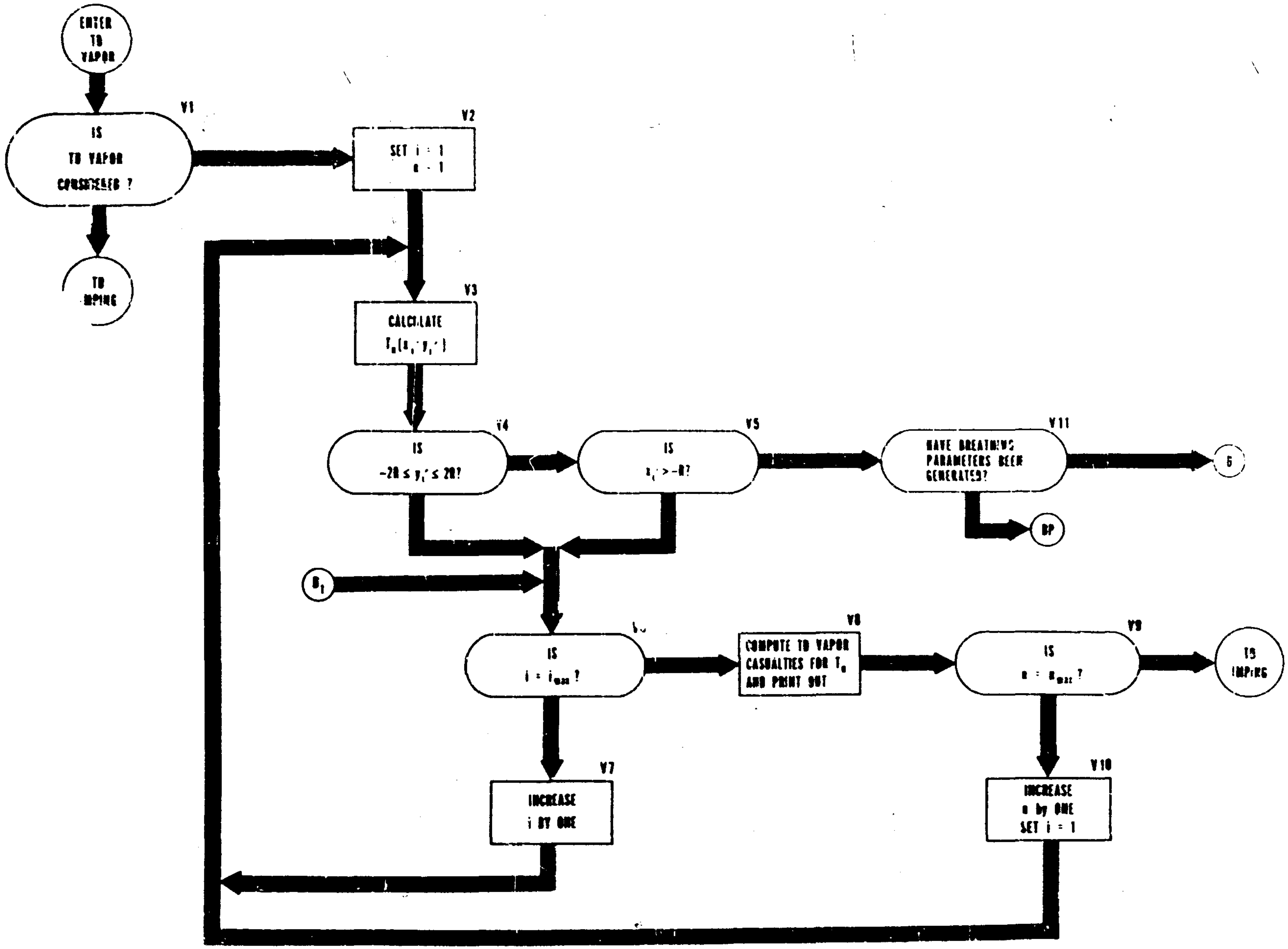
4. SPLASH



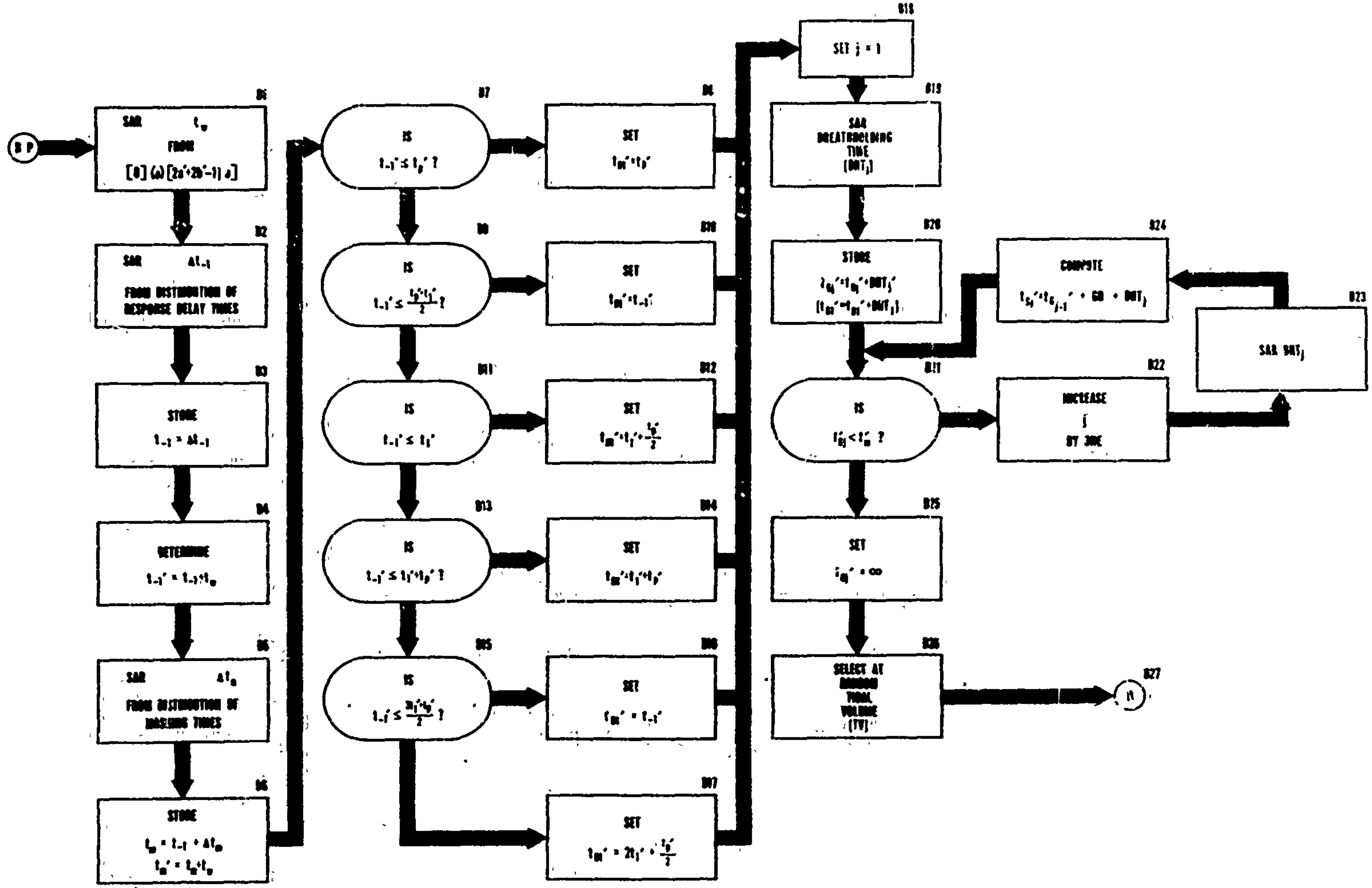
STORED FOR SPLASH

k_{max}
λ_k for all k
a_k for all k
b_k for all k
c_k for all k

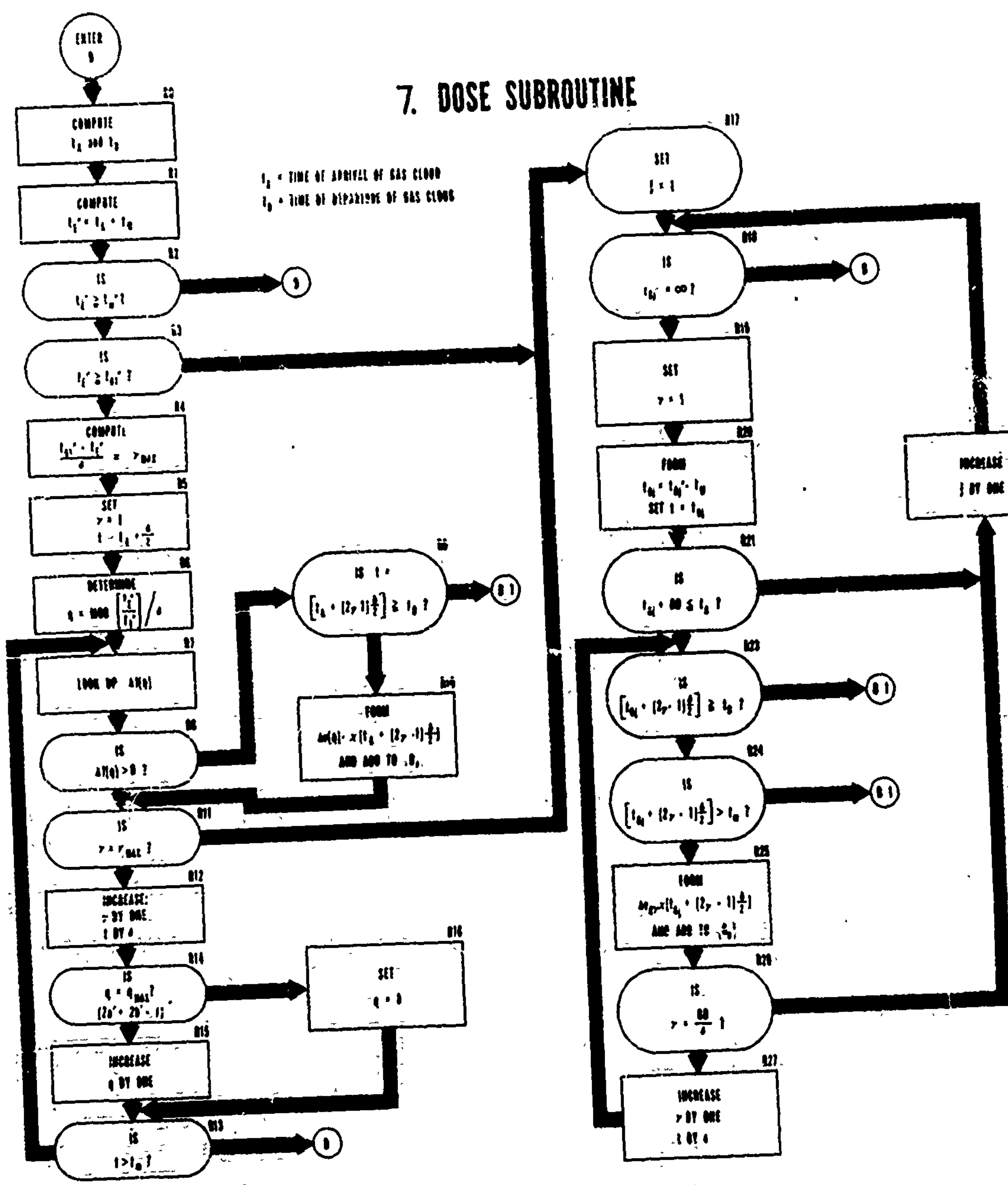
5. TIME DEPENDENT VAPOR



C. BREATHING PARAMETER SUBROUTINE



7. DOSE SUBROUTINE



III. DESCRIPTION OF THE FLOW CHARTS

3.1 This section presents a list of symbols used in the flow charts and a step-by-step description of each operation. Each explanation is keyed to a corresponding block on the flow charts by the use of block numbers.

3.2 Although Flow Chart 1 (the Master Flow Chart) provides for determination of time dependent impingement (downwind impingement), there is no flow chart or program at present to carry out this determination. This feature of the model has not yet been developed, but is expected to be added early in the coming contract year. The modular nature of the model makes this addition quite simple.

3.3 Since this model is, by design, a Monte Carlo simulation, it is of interest to summarize the variables whose values are determined by sampling from probability distributions. In the current form of the simulation, the following probability distributions are used.

a. Target element location error (assumed symmetrical in x and y [1/]).

b. Battery-volley centroid x-coordinate error.

c. Battery-volley centroid y-coordinate error.

[1/] The assumed symmetry in x and y makes the detec''on process correspond to aerial photographic reconnaissance. If, for example, artillery observer reconnaissance were used, there would be different distributions for x and for y.

d. Individual-piece x-coordinate error.

e. Individual-piece y-coordinate error.

f. Response delay time.

g. Masking time.

h. Duration of breathhold.

i. Tidal volume.

3.4 With further development it will be possible to sample values of additional quantities such as source strength, stability parameter, and wind speed.

3.5 In addition to the sampling of probability distributions just referred to, there are currently three other random determinations:

a. At the start there is a random selection of three target elements from the array of target elements (see Block T-1).

b. In the fragmentation routine a target may be determined to be in a zone of potential fragmentation casualty. When this is the case a random number is compared with the probability of casualty for the particular zone. If the random number is less than or equal to the probability the target is a casualty (see Block B-16).

c. In the breathing parameter subroutine the target's phase in the breathing cycle is selected at random.

3.6 Currently, the chemical dose-response relation is not used in the program which has been written. The final output of a computer run is the total chemical dose received by each target element. It is a relatively simple matter to extend the program so that a Monte Carlo determination of degree of casualty can be made following the determination of the chemical dose.

NOTATION FOR FLOW DIAGRAM

3.7 The notation utilized in the flow diagram is defined under the following classifications:

1. Coordinate Notation
2. Targeting and Delivery Notation

3. Fragmentation Notation

4. Splash Notation

5. Breathing Parameter Notation

6. Dosage Notation

Coordinate Notation

(x_i, y_i)	Location of impact of the i^{th} shell
(x_n, y_n)	Location of the n^{th} target
(x_α, y_α)	Location of one of the targets observed
(x_α', y_α')	Estimated location of one of the targets observed
$(\bar{x}', \bar{y}')$	Estimated centroid of target observed; location at which battery fires
$(\bar{x}, \bar{y})$	Location of impact centroid
(x'', y'')	Errors in firing of individual guns
(x_i^w, y_i^w)	Location of impact of i^{th} shell in wind coordinates
(x_n^w, y_n^w)	Location of n^{th} target in wind coordinates
(x', y')	Distance downwind and crosswind from impact to target.

Targeting and Delivery Notation

i	Impact index; $i = 1, 2, \ldots, i_{MAX}$.
L	Location of impact of shell fired from left-most gun if there were no individual piece firing error.
n	Target index; $n = 1, \ldots, n_{MAX}$.
r'	The largest integer obtained by dividing the number 1000 by n_{MAX}.
ζ	The distance between aiming points of each gun.
θ	The direction of the wind in degrees measured counterclockwise from the x-axis.

Fragmentation Notation

d_{ni}	Distance of the n^{th} target from the i^{th} impact.
β_{ni}	Cotangent of the angle between the x-axis and the line between the impact and the target.

Δ	A small distance from the impact in which a maximum casualty occurs.
ρ_{MAX}	Maximum range of fragmentation.
τ	Sector in which fragmentation occurs; $\tau = 1,2,\ldots,\tau_{MAX}$
ξ	Denotes varying radii as measured from the impact points; $\xi = 1,2,\ldots,\xi_{MAX(\tau)}$
$\alpha_{\tau 1}$	Cotangent of the angle between side 1 of the τ^{th} sector and the x-axis.
$\alpha_{\tau 2}$	Cotangent of the angle between side 2 of the τ^{th} sector and the x-axis.
$r_{\tau\xi}$	The radii for the τ^{th} sector.
$Z_{\tau\xi}$	The zone formed by the τ^{th} sector and the radii, $r_{\tau\xi}$ and $r_{\tau,\xi-1}$

Splash Notation

k	Index for denoting ellipses of different sizes; $k = 1,2,\ldots,k_{MAX}$.
a_k	Distance from the center of the k^{th} ellipse to the edge of the ellipse along the major axis.
b_k	Distance from the center of the k^{th} ellipse to the edge of the ellipse along the minor axis.
c_k	Distance from the center of the k^{th} ellipse to each focus.
λ_k	Distance from the center of the k^{th} ellipse to the impact point.
ρ_{nik}	Sum of the distances of the target from each of the two foci for the k^{th} ellipse.

Breathing Parameter Notation

j	Gasp index; $j = 1,2,\ldots,j_{MAX}$.
BHT_j	Time for j^{th} breathhold plus exhale time.
GD	Gasp Duration (duration of inhale portion of gasp).
q	Indexes an interval of width δ in the breathing cycle; $q = 1,2,\ldots t_1'/\delta$

R	Burst radius of shell.
δ	Increment of time in breathing cycle used for numerical integration.
t_A	Time of arrival of gas cloud relative to time of impact.
t_D	Time of departure of gas cloud relative to time of impact.
Δt_m	Masking time (in increments of δ).
t_m	Time at which masking is completed relative to time of impact.
t_w	The time relative to the beginning of a man's breathing cycle at impact time.
t_{-1}	Response delay time (increments of δ).
t	Time relative to time of impact.
t_E'	Time when gas cloud arrives relative to beginning of the man's breathing cycle.
t_{Gj}'	Time at which the j^{th} "gasp" begins relative to beginning of the man's breathing cycle
t_{H1}'	Time at which breathholding first begins relative to beginning of breathing cycle.
t_m'	Time at which masking is completed relative to beginning of the man's breathing cycle.
t_p'	The time relative to the beginning of the man's cycle when the peak of the breathing cycle is obtained.
t_1'	The length of the breathing cycle.
t_{-1}'	Time relative to beginning of the breathing cycle when the man can respond to the impact.
TV	Tidal volume of air intake.
a,b	Integers; the duration of inhale is $2a\delta$; the duration of exhale is $2b\delta$; $t_p' = 2a\delta$; $t_1' - t_p' = 2b\delta$.

<u>Dosage Notation</u>

B_n	Location for accumulating steady breathing dose.

G_n	Location for accumulating gasp dose.
q	Intervals in breathing cycle; $q = 1, 2, \ldots t_1'/\delta$.
$\bar{u}$	Wind speed.
α	Stability parameter.
γ	Index used for numerical integration.
$\Delta f(q)$	Fraction of TV inhaled during the q^{th} interval.
$\Delta v(q)$	Volume of air intake during the q^{th} interval.
$\chi(t, x'y')$	Concentration of cloud at time t when target is x' and y' from impact.

DESCRIPTION OF FLOW CHARTS—ACC-2, 20 JUNE 1961

Master Flow Chart

BLOCK M-1. Enter Pilot Model

The input elements—the constants and probability distributions necessary for the battle-field model considered—are entered into the computer.

BLOCK M-2. Input Target Elements

The (x, y) coordinates of the targets are entered into the computer. The number of target elements varies from 3 to 100.

BLOCK M-3. Select Aiming Point

The point at which the guns are to be fired is determined as a consequence of randomly selecting three of the actual target elements (errors associated with target acquisition are considered).

BLOCK M-4. Compute Impact Points

The coordinates of each of the impact points are determined. Two sets of coordinate systems are used throughout the simulation:

1. a regular coordinate system and
2. a wind coordinate system.

BLOCK M-5. Is Frag Considered?

This block makes consideration of fragmentation effects optional.

BLOCK M-6. Compute Frag Effects/Casualties

This block determines the fragmentation effects.

BLOCK M-7. Is Splash Considered?

This block makes consideration of splash effects optional.

BLOCK M-8. Compute Splash Dose/Casualties

This block determines the splash effects.

BLOCK M-9. Is Time Dependent Vapor Considered?

This block makes consideration of time dependent vapor effects optional.

BLOCK M-10. Compute Vapor Dose/Casualties

This block determines the results of the time dependent vapor.

BLOCK M-11. Is Time Dependent Impingement Considered?

This block makes consideration of time dependent impingement effects optional.

BLOCK M-12. Compute Impingement Dose/Casualties

This block determines the results of the time dependent impingement.

BLOCK M-13. Cumulate Chemical Dose

This block determines the total chemical dosage obtained from the splash effect, the time-dependent vapor effect, and the time-dependent impingement effect.

BLOCK M-14. Compute Net Chemical Casualties

This block determines the total chemical casualties as a result of the chemical dose obtained in Block M-13.

BLOCK M-15. Is There to Be Another Replication?

This block tells the computer the number of times to perform the simulation. If there is to be another replication, the sequence of events that began in Block M-1 is repeated.

BLOCK M-16. Stop

When all replications for the specified input elements are completed, the computer stops.

The blocks as generally described in the Master Flow Chart will be described in more detail in the following pages.

Targeting and Delivery

BLOCK T-1. Select at Random Three Targets

From the n_{MAX} target elements entered into the machine, three target elements are selected at random. The random process for selecting these three targets is as follows:

1. The largest integer, r', that can be obtained by dividing the number 1000 by n_{MAX} is determined.
2. Using the interval width of r', n_{MAX} intervals are constructed.
3. There is an interval which corresponds to each of the target elements, $1, 2, \ldots, n_{MAX}$. (Any remainder, 1000 - $(r')(n_{MAX})$, as a consequence of dividing n_{MAX} into 1000, is not a usable random number.)
4. The first target element is obtained by determining the correspondence between the interval in which the first random number falls and the input target elements. If the random number selected is greater than $(r')(n_{MAX})$, another random number is chosen.
5. After the first target is selected, there are two intervals which are not usable, the interval corresponding to the first target selected and the interval greater than $(r')(n_{MAX})$.
6. This process is repeated until three different target elements have been selected.

BLOCK T-2. Calculate the Centroid

The centroid of the three targets selected at random is calculated by the following equations:

$$\bar{x} = \sum_{\alpha=1}^{3} x_\alpha/3 \qquad \bar{y} = \sum_{\alpha=1}^{3} y_\alpha/3 \quad ,$$

BLOCK T-3. Estimate Coordinates of Randomly Selected Targets

The x and y errors involved in the estimates of the three target elements are determined from a probability distribution of errors. In the current version of the model the variance of the x errors is identical with the variance of the y errors. These variances can be different. The estimated coordinates of the three randomly selected targets are obtained by adding these x and y errors to the actual x and y coordinates of the three randomly selected target elements, respectively.

BLOCK T-4. Calculate Estimated Centroid

The estimated centroid is the point at which the center of the volley is aimed. This point is obtained from the estimated coordinates obtained in Block T-3 by the following two equations:

$$\bar{x} = \sum_{\alpha=1}^{3} x_\alpha'/3 \qquad \bar{y} = \sum_{\alpha=1}^{3} y_\alpha'/3 \quad ,$$

BLOCK T-5. Print Targeting Information

The following targeting information is printed:

1. The coordinates of each of the randomly selected target elements (determined in Block T-1).
2. The actual centroid of the three randomly selected elements (determined in Block T-2).
3. The coordinates of the three estimated target elements (determined in Block T-3).
4. The estimated centroid for firing (determined in Block T-4).

BLOCK D-1. Select Impact Centroid

The impact centroid is determined by adding the x and y errors in battery firing to the coordinates of the estimated centroid (calculated in Block T-4). The x and y errors are obtained from a probability distribution of errors in the impact centroid from the estimated centroid. The variance of the x errors is not equal to the variance of the y errors. This block takes into account the artillery battery delivery error.

BLOCK D-2. Select Impact Errors for Each Impact

The x and y errors for each impact point about the point at which the gun was fired are obtained from a probability distribution of individual piece firing errors. An x and y error is selected for each of the i_{MAX} impact points.

BLOCK D-3. Computer Impact Coordinates in Regular (x, y) System

Figure 1 illustrates the computation of the impact coordinates. The x-coordinate of the impact point of the shell fired from the left-most gun would be located a distance of $(i_{MAX}-1)(\zeta/2) = L$ from the x-coordinate of the impact centroid (determined in Block D-1) if there were no individual piece firing error. The remaining x-coordinates of the points at which the shells would strike if there were no individual piece firing errors are

$$L + \zeta \ , \ L + 2\zeta, \ldots, L + (i_{MAX}-1)\zeta.$$

By adding the x-errors as obtained in Block D-2 to $L + \zeta$, $L + 2\zeta, \ldots,$ $L + (i_{MAX}-1)\zeta$, the actual x-coordinates of the individual impact points are determined. The actual y-coordinates for each of the individual impact points are determined by adding the y-errors, as obtained in Block D-2, to the y-coordinates of the impact centroid, as obtained in Block D-1.

BLOCK D-4. Compute Wind Coordinates of Impact Points

The impact coordinates are determined when the regular x and y axes are rotated by the number of degrees corresponding to the wind direction. The following equations are used:

$$x_i^W = x_i \cos\theta + y_i \sin\theta$$

$$y_i^W = y_i \cos\theta - x_i \sin\theta$$

where x_i and y_i are the impact coordinates in the regular coordinate system (where the x axis is parallel to FEBA and the y axis is perpendicular to FEBA), x^W and y^W are the coordinates in the wind coordinate

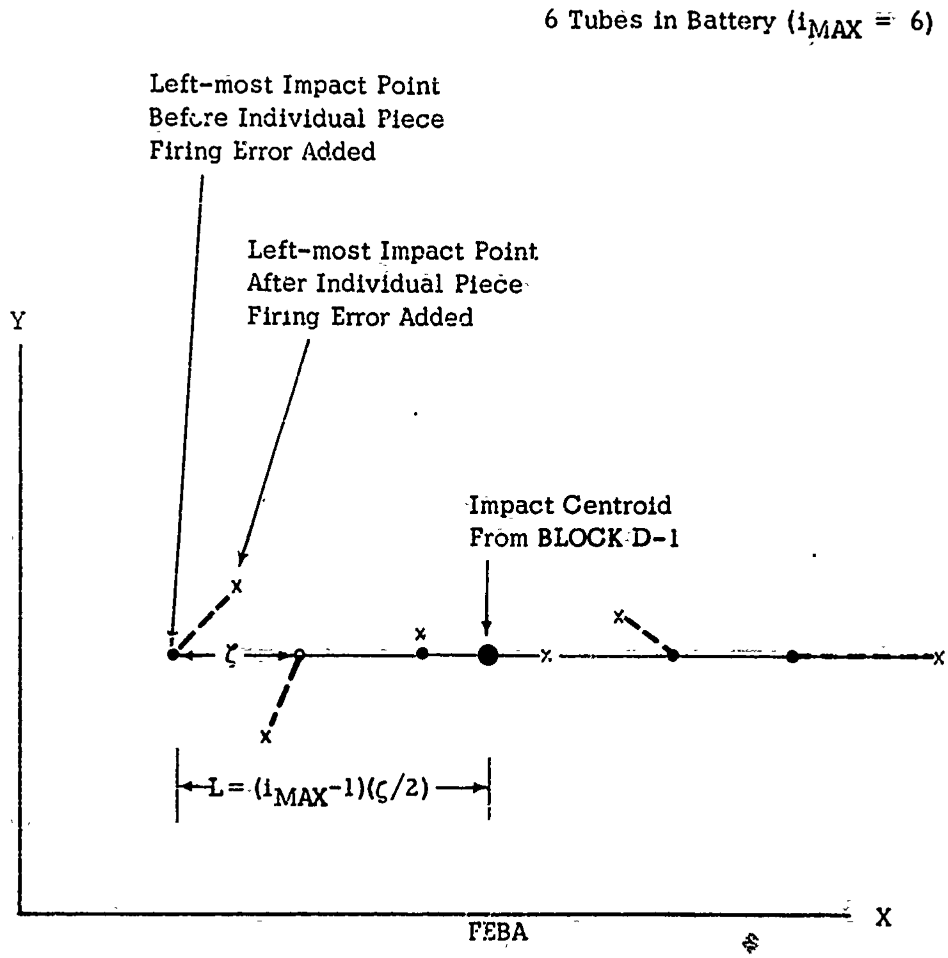


FIGURE 1. COMPUTATION OF THE IMPACT COORDINATES

system, and θ is the wind direction in degrees with respect to the x axis (the x^W axis is parallel to wind direction).

BLOCK D-5. Print Delivery Information

The impact centroid and the impact points for each of the shells are printed both in the regular coordinate system and also in the wind coordinate system.

Fragmentation

BLOCK F-1. Is Frag Considered?

This block makes consideration of fragmentation effects optional.

BLOCK F-2. Set $i = 1$ and $n = 1$

In this block i and n are set equal to one.

BLOCK F-3. Is $y_i \leq y_n$?

See Figure 2. The coordinates of the impact point are denoted by (x_i, y_i), and the coordinates of the target are denoted by (x_n, y_n). Whenever the y coordinate of the impact point, y_i, is greater than the coordinate of the target, y_n, the target is not hit.

BLOCK F-4. Compute Distance of Target from Impact Point

For economy in computer operation, the squares of distances are computed and compared, rather than the actual distances. The distance squared of the target from the impact point is computed by the following equation:

$$d_{ni}^2 = (x_n - x_i)^2 + (y_n - y_i)^2 \quad .$$

BLOCK F-5. Is $d_{ni}^2 \leq \rho_{MAX}^2$?

There is a distance, ρ_{MAX}, from the impact point beyond which fragmentation effects are negligible. If fragmentation effects are negligible, the computer returns to Block F-21. If the target is within ρ_{MAX}, the severity

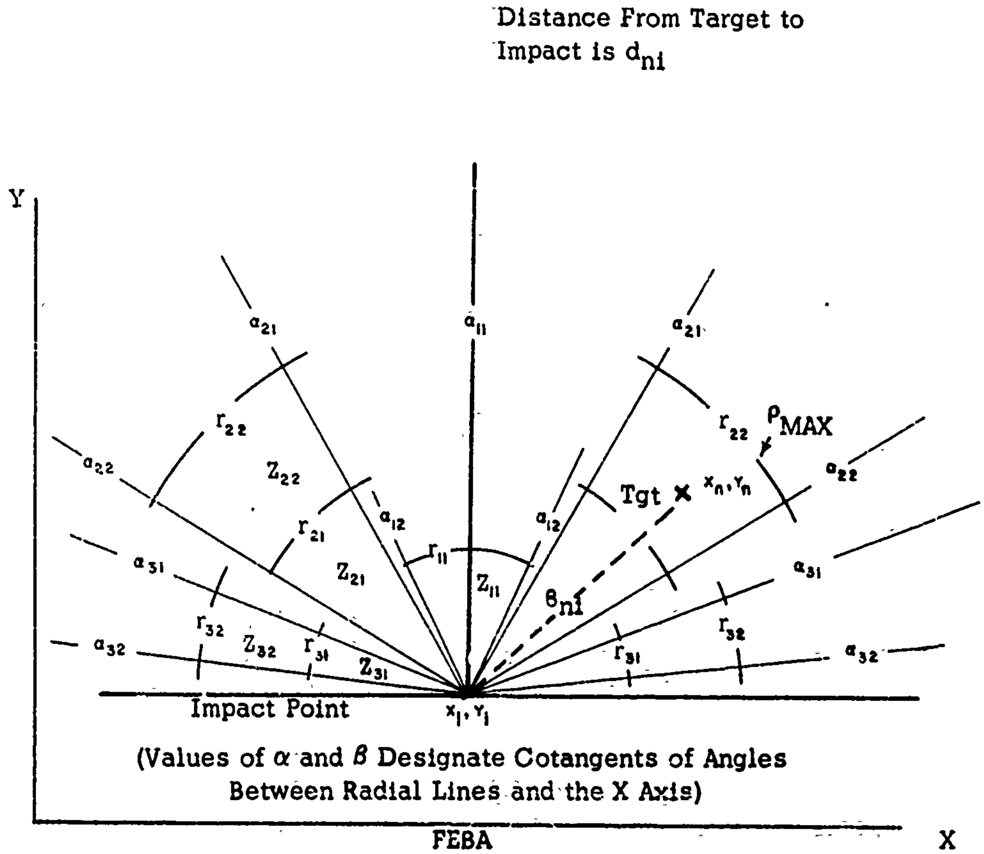


FIGURE 2. FRAGMENTATION PATTERN

of fragmentation effects is computed in the remaining blocks.

BLOCK F-6. Is $d_{ni}^2 \leq \Delta^2$?

It is possible that the distance between the target and the impact point is so small, Δ, that a direct hit is obtained. If a direct hit is obtained, the occurrence of such a hit is stored.

BLOCK F-7. Store Maximum Casualty

If a direct hit has occurred, maximum casualty is stored.

BLOCK F-8. Is $y_i = y_n$?

In this block, and also Block F-9, and F-10, attention is focused on determining the cotangent, β_{ni}, of the angle between the x axis and the line connecting the impact point and the target. If the impact point and the target are the same distance from the x-axis, $y_i = y_n$, β_{ni}, is infinite.

BLOCK F-9. Set $\beta_{ni} = \infty$

Since the computer is not equipped to handle a number as large as infinity, special handling must take place. The location which contains β_{ni} is set equal to 9.9×10^{40} to indicate that β_{ni} is equal to infinity.

BLOCK F-10. Compute β_{ni}

Since this cotangent is less than infinity, the computation for β_{ni} is obtained by using the following equation:

$$\beta_{ni} = \left| \frac{x_n - x_i}{y_n - y_i} \right| .$$

BLOCK F-11. Set $\tau = 1$

τ denotes the sector in which fragmentation effects occur. (See Figure 2.) With $\tau = 1$, a determination will be made to see if the target is in the first sector. In general, there are τ_{MAX} sectors ($\tau = 1, 2, \ldots, \tau_{MAX}$).

BLOCK F-12. Set $\xi = 1$

The Greek letter ξ is used as a subscript to denote varying radii as measured from the impact point. In general, ξ is equal to $1, 2, \ldots, \xi_{MAX(\tau)}$. The radii for the τ^{th} sector are denoted by $r_{\tau 1}, r_{\tau 2}, \ldots, r_{\tau \xi MAX(\tau)}$.

BLOCK F-13. Is $\alpha_{\tau 1} \leq \beta_{ni}$?

As shown in Figure 2, $\alpha_{\tau 1}$ is the cotangent of the angle between side one of the τ^{th} sector and the x axis. If $\alpha_{\tau 1}$ is not less than or equal to β_{ni}, the target is not in the τ^{th} sector. In this case the machine returns to F-21 and the next impact is considered. If $\alpha_{\tau 1}$ is less than or equal to β_{ni}, a check is made in Block F-14 against side two of the τ^{th} sector.

BLOCK F-14. Is $\beta_{ni} \leq \alpha_{\tau 2}$?

The notation $\alpha_{\tau 2}$ is used to denote the cotangent of the angle between side two of the τ^{th} sector and the x axis. If β_{ni} is equal to or less than $\alpha_{\tau 2}$, the target is in the τ^{th} sector.

BLOCK F-15. Is $d_{ni}^{\,2} > r_{\tau \xi}^{\,2}$?

Since ξ is initially set to one, a no answer means that the target lies within radius one of sector τ. This region is denoted zone $Z_{\tau 1}$. If the answer is yes, ξ is increased until the zone is determined.

BLOCK F-16. Calculate Fragmentation Effects for Zone $Z_{\tau \xi}$

Corresponding to each zone, $Z_{\tau \xi}$, there is a probability that a fragmentation casualty occurs. A casualty occurs if a random number is equal to or less than the probability of a casualty. The number of casualties for each target element is accumulated over all impacts.

BLOCK F-17. Is ξ equal to $\xi_{MAX(\tau)}$?

If ξ is equal to $\xi_{MAX(\tau)}$, the target element is in an area free from fragmentation. The sequence of operations in F-21 follows and the effects of the next impact are considered. If ξ is not equal to $\xi_{MAX(\tau)}$, then the target element may be within a radius corresponding to a greater value of ξ and the next larger radius is considered.

BLOCK F-18. Increase ξ by one

For the same sector of τ, ξ is increased to the next radius, $\xi + 1$. The sequence of operations in Block F-15 follows.

BLOCK F-19. Is $\tau = \tau_{MAX}$?

If τ is equal to τ_{MAX}, then the target element is outside the last fragmentation sector and fragmentation effects are negligible. If τ is not equal to τ_{MAX} (that is, τ is less than τ_{MAX}), the next τ sector is considered.

BLOCK F-20. Increase τ by one

Since τ is less than τ_{MAX}, not all sectors have been considered. Therefore, the next sector is considered by increasing τ by one. After increasing τ, the sequence of operations in Block F-12 follows.

BLOCK F-21. Clear d_{ni}^2 and β_{ni}

Since the next impact is to be considered, certain storage locations must be cleared.

BLOCK F-22. Is $i = i_{MAX}$?

If i is equal to i_{MAX}, all impact points for this target element have occurred and a print-out follows. If i is not equal to i_{MAX} (that is, i is less than i_{MAX}), further impact points are to be considered.

BLOCK F-23. Increase i by one

The next impact is considered by increasing i by one.

BLOCK F-24. Print Out

The results of the impacts on the target elements are printed out. Specifically, the number of times in which the target element is a casualty is printed out.

BLOCK F-25. Set $i = 1$

The effects of each of the i_{MAX} impacts on the next target element are considered, again beginning with the first impact.

BLOCK F-26. $\underline{\text{Is } n = n_{MAX}?}$

If n is equal to n_{MAX}, the effects of all impacts on each of the target elements have been evaluated and printed; therefore, the computer enters the routine for evaluating the effects of splash. If n is not equal to n_{MAX} (that is, n is less than n_{MAX}), the effects on all of the target elements have not been evaluated.

BLOCK F-27. Increase n by One

The value of n is increased by one so that the fragmentation effects on the next target element may be evaluated.

Splash

BLOCK S-1. Is Splash Considered?

This block makes consideration of splash effects optional.

BLOCK S-2. $\underline{\text{Set } i = 1,\ n = 1,\ k = k_{MAX}}$

This block initializes values for the three parameters i, n, and k. The letter k, is used as an index for denoting ellipses of different sizes; as k increases, the size of the ellipse increases.

> Note to BLOCKS S-3, S-4, S-5, and -6:
> In Blocks S-3, S-4, S-5, and S-6, the index k has only the value k_{MAX} so that only the largest splash region is under consideration. The largest splash ellipse, k_{MAX}, is approximated by a circumscribing rectangle. If the target element lies inside this rectangle, further tests starting with BLOCK S-7 are employed. These tests determine the annular region between ellipses in which the target element lies.

BLOCK S-3. $\underline{\text{Is } y_n > y_i + \lambda_k + b_k?}$

This block determines whether the target element is too far downrange from the point of impact.

BLOCK S-4. $\underline{\text{Is } y_n < y_i + \lambda_k - b_k?}$

This block determines whether the target element is too far uprange from the point of impact.

BLOCK S-5. $\underline{\text{Is } x_n > x_i + a_k?}$

This block determines whether the target element is too far to the right of the point of impact.

BLOCK S-6. $\underline{\text{Is } x_n < x_i - a_k?}$

This block determines whether the target element is too far to the left of the point of impact.

BLOCK S-7. $\underline{\text{Compute } \rho_{nik}}$

Refer to Figure 3. The symbol ρ_{nik} is used to denote the sum of the distances of the target from each of the two foci of the k^{th} ellipse and c_k is used to denote the distance from the center of the k^{th} ellipse to each focus. The formula for ρ_{nik} is as follows:

$$\rho_{nik} = \sqrt{(x_n - x_i)^2 + (y_n - y_i + c_k - \lambda_k)^2} + \sqrt{(x_n - x_i)^2 + (y_n - y_i - c_k - \lambda_k)^2} .$$

BLOCK S-8. $\underline{\text{Is } \rho_{nik} > 2a_{k_{MAX}}?}$

If $\rho_{nik} > 2a_{k_{MAX}}$, then the target element is not within the $k_{MAX}{}^{th}$ ellipse and the next impact point will be considered. If ρ_{nik} is less than $2a_{k_{MAX}}$, then the target element is within the $k_{MAX}{}^{th}$ ellipse. The subsequent blocks determine the lowest numbered ellipse which includes the target element.

BLOCK S-9. $\underline{\text{Is } k = 1?}$

If k is equal to one, the target element is within the first ellipse or region one. Proceed to Block S-13. If k is not equal to one, proceed to Block S-10.

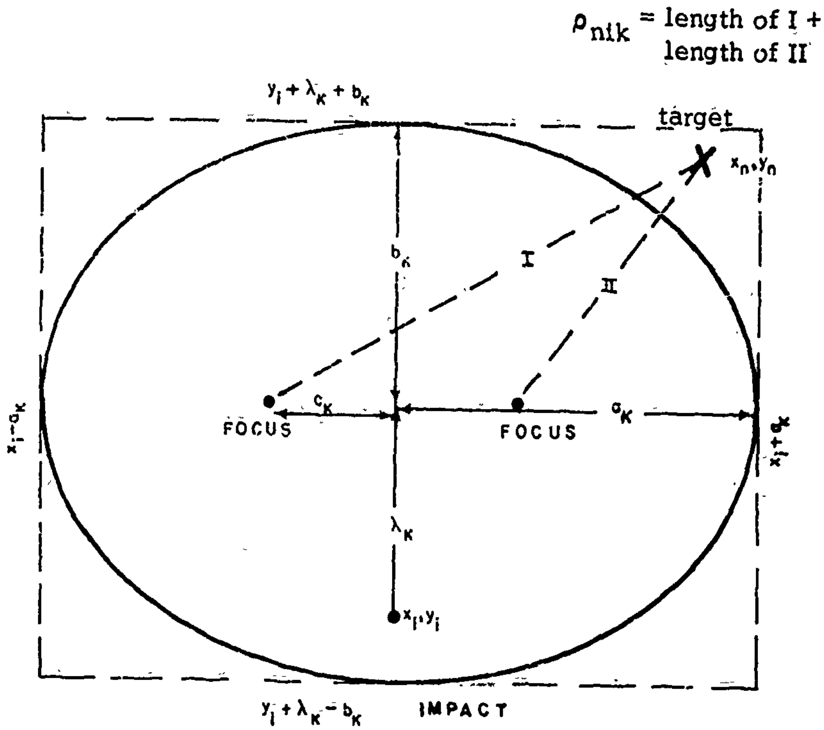


FIGURE 3. SPLASH PATTERN SHOWING K_{MAX} ELLIPSE AND CIRCUMSCRIBED RECTANGLE

BLOCK S-10. Decrease k by One and Compute ρ_{nik}

The value of ρ_{nik} for the next smaller ellipse is computed.

BLOCK S-11. Is $\rho_{nik} > 2a_k$?

If ρ_{nik} is greater than $2a_k$, then the target element is not within the k^{th} ellipse, but is within the $k + 1^{th}$ ellipse. An inquiry is made about the target being in the k^{th} ellipse only if the target is within the $k + 1^{th}$ ellipse. If ρ_{nik} is less than $2a_k$, then the target element is within the k^{th} ellipse. Therefore, the target element is in one of the following regions: k, k - 1,..., 2, 1.

BLOCK S-12. Determine Splash for Region k + 1

Since it has been ascertained that the target element is within the $k + 1^{th}$ region, the corresponding amount of splash is determined.

BLOCK S-13. Determine Splash for Region 1

As a result of the logic in Block S-9, it is determined that the target element is within region one and the corresponding quantity of splash is determined.

BLOCK S-14. Set $k = k_{MAX}$

Since the location of the target element with respect to the splash regions has been determined for this impact, the value of k is reset to k_{MAX} for use in determining the location of the target element with respect to the next impact.

BLOCK S-15. Is $i = i_{MAX}$?

If i is equal to i_{MAX}, all impacts have been considered with reference to a specific target element. If i is not equal to i_{MAX} (that is, i is less than i_{MAX}), then the next impact is to be considered.

BLOCK S-16. Increase i by One

The next impact is considered by increasing i by one.

BLOCK S-17. <u>Is $n = n_{MAX}$?</u>

If n is equal to n_{MAX}, all target elements have been evaluated for all impacts and the computer will proceed to the evaluation of time-dependent vapor. If n is not equal to n_{MAX} (that is, n is less than n_{MAX}), the next target element is considered.

BLOCK S-18. <u>Increase n by one, set i = 1</u>

In order to ascertain the effects of the splash upon additional target elements for each of the impacts, n is increased by one and i is set equal to one.

<u>Time Dependent Vapor</u>

BLOCK V-1. <u>Is Time Dependent Vapor Considered?</u>

This block makes consideration of time dependent vapor effects optional.

BLOCK V-2. <u>Set i = 1; n = 1</u>

This block initializes values for the parameters i and n.

BLOCK V-3. <u>Calculate x' and y' distances from impact to target</u>

(See Figure 4.) This block calculates the x' distance and y' distance (wind coordinates) between the target and the impact. The equations used for this calculation are as follows:

$$x' = x_n^W - x_i^W \qquad \text{and} \qquad y' = y_n^W - y_i^W \quad .$$

BLOCK V-4. <u>Is $-2R \le y' \le 2R$?</u>

This block determines whether or not the target is within permissible bounds on the y^W axis. R is the burst radius of the cloud. If $-2R \le y' \le 2R$, then the target may be subject to the vapor cloud effects. If y' is less than -2R or if y' is greater than +2R, the vapor cloud has no effect upon the target.

BLOCK V-5. <u>Is $x' > -R$?</u>

If x' is greater than -R, then the vapor cloud may affect the target. If x' is not greater than -R (that is, x' is less than -R), the vapor cloud has no effect upon the target.

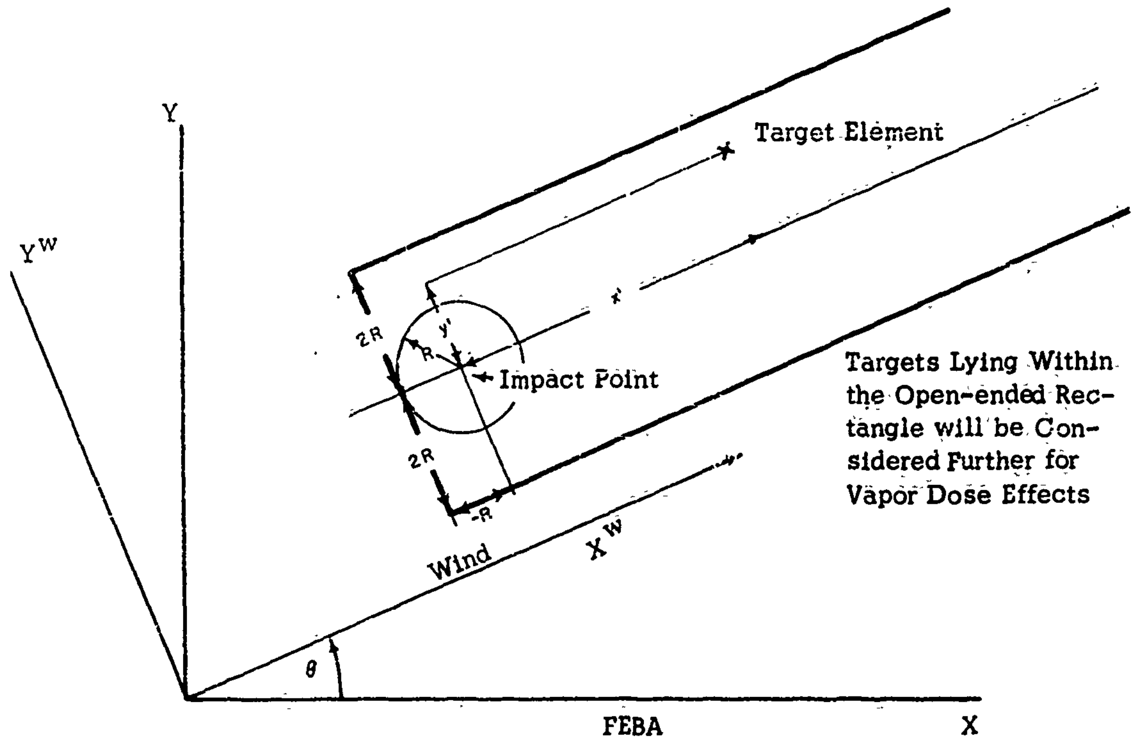


FIGURE 4. RELATIVE COORDINATES IN THE WIND-COORDINATE SYSTEM

BLOCK V-6. Is $i = i_{MAX}$?

If i is equal to i_{MAX}, the effects of the vapor cloud for each of the impacts have been considered upon a specific target. If i is not equal to i_{MAX} (that is, i is less than i_{MAX}), then the effects of additional impacts must be considered for the target under consideration.

BLOCK V-7. Increase i by One

The value of i is increased by one, so that the vapor effects of the next impact upon the target element may be evaluated.

BLOCK V-8. Compute Time Dependent Vapor Casualties for a Target Element and Print Out

The effects of the various dosages of vapor, which are obtained by breathing and gasping, are summarized for each man and printed out. Also, there is a print-out for each man which gives the number of times, if any, in which he received a casualty-producing dose.

BLOCK V-9. Is $n = n_{MAX}$?

If n is equal to n_{MAX}, the effects of each of the vapor clouds for all impacts upon each of the target elements have been evaluated and pertinent information printed out; therefore, the computer enters the routine for evaluating the effects of the time dependent impingement. If n is not equal to n_{MAX} (that is, n is less than n_{MAX}), the next target element is considered.

BLOCK V-10. Increase n by One; set $i = 1$

The value of n is increased by one, so that the vapor cloud effects upon the next target element will be evaluated for each of the impacts.

BLOCK V-11. Have Breathing Parameters Been Generated?

If the breathing parameters have been generated, the computer enters the dose subroutine. If the breathing parameters have not been generated, the computer enters the breathing parameter subroutine.

Breathing Parameters

BLOCK B-1. Select at Random t_w.

The random variable, t_w, is the time relative to the beginning of a man's breathing cycle at impact time. The variable t_w is expressed as an integral number of intervals of width δ. The inhale time is $2a\delta$; the exhale time is $2b\delta$; a and b are integers.

BLOCK F-2. Select at Random Response Delay Time

This is the time which the target element takes before reacting to the impact. This delay time is expressed as an integral number of intervals of width δ.

BLOCK B-3. Store Response Delay Time

The value of the response delay time, t_{-1}, is stored. Note: In the determination of the breathing parameters, it is convenient to have two time scales. One time scale is measured from impact time. The other scale is measured from the beginning of the breathing cycle of a particular target element. This time scale is denoted by time symbols containing a prime (that is, t'). A time scale in terms of the impact is shifted t_w time units from the other time scale.

BLOCK B-4. Determine t_{-1}'

Symbol t_{-1}' is used to denote the time relative to the beginning of the breathing cycle when the target element begins to respond to the impact. This time value is obtained as follows:

$$t_{-1}' = t_{-1} + t_w.$$

BLOCK B-5. Select at Random Masking Times

The time required for each target element to mask, Δt_m, is determined from a distribution of masking times, which are expressed as an integral number of intervals of width δ.

BLOCK B-6. Compute and Store Masking Times

The times at which masking occurs is determined for both time scales from the following equations:

$$t_m = t_{-1} + \Delta t_m \text{ and}$$

$$t_m' = t_m + t_w.$$

Breathhold Rules

There are five blocks which are used to determine the time at which the target element begins the breathhold procedure. See Figure 5. The rules which are used in establishing the time for breathhold are as follows:

(1) If at the end of the response delay time, a man is in the inhale portion of the cycle, he will continue inhaling until he reaches the cycle peak, t_p'.

(2) If at the end of the response delay time, the man is in the exhale portion and has exhaled less than 50% of the volume of air, he will begin the breathhold immediately.

(3) If at the end of the response delay time, the man is in the exhale portion of the cycle and has exhaled more than 50% of the volume of air, he will continue exhaling and inhale 50% of the normal volume before breathhold begins.

BLOCK B-7. Is $t_{-1}' \le t_p'$?

If t_{-1}' is equal or less than t_p' then the man was in the inhale portion of the cycle. Otherwise, the man was not in the inhale portion of the first cycle.

BLOCK B-8. Set $t_{H1}' = t_p'$

In accordance with rule one, as stated above, the man begins breathhold at the peak of the cycle, if he is in the inhale portion at the end of the response delay time.

BLOCK B-9. Is $t_{-1}' \le \frac{t_p' + t_1'}{2}$?

If t_{-1}' is equal to or less than $\frac{t_p' + t_1'}{2}$, then the man was in the exhale portion of the cycle and less than 50% of all the volume of intake air had been exhaled. If t_{-1}' is not equal to or less than $\frac{t_p' + t_1'}{2}$, the man is not in the first half of the exhale portion of the cycle in which the impact occurred.

BLOCK B-10. Set $t_{H1}' = t_{-1}'$

In accordance with the breathhold rules, when the man is in the exhale portion of the cycle and less than 50% volume has been exhaled, he begins breathhold immediately.

BLOCK B-11. Is $t_{-1}' \leq t_1'$?

If t_{-1}' is equal to or less than t_1', then the man was in the exhale portion of the cycle and more than 50% of volume had been exhaled. If t_{-1}' is not equal to or less than t_1', then the man was not in the breathing cycle in which the impact occurred.

BLOCK B-12. Set $t_{H1}' = t_1' + \frac{tp'}{2}$

In accordance with the third breathhold rule, when the man is in the exhale portion of the cycle and more than 50% of his intake volume has been exhaled, the time at which he begins breathhold occurs after he has completed the exhale and half of the next inhale.

BLOCK B-13. Is $t_{-1}' \leq t_1' + t_p'$?

If t_{-1}' is equal to or less than $t_1' + t_p'$, then the man is in the inhale portion of the second cycle. If t_{-1}' is not equal to or less than t_p', then the man is not in the inhale portion of the second cycle.

BLOCK B-14. Set $t_{H1}' = t_1' + t_p'$

In accordance with the first breathhold rule, the time at which breathhold begins is at the peak of the inhale cycle.

BLOCK B-15. Is $t_{-1}' \leq \frac{3t_1' + t_p'}{2}$?

If t_{-1}' is equal to or less than $\frac{3t_1' + t_p'}{2}$, the man was in the exhale portion of the second cycle and less than 50% of volume had been exhaled. If t_{-1}' is not equal to or less than $\frac{3t_1' + t_p'}{2}$, then the man is not in the first half of the exhale portion of the second cycle since the impact occurred.

BLOCK B-16. Set $t_{H1}' = t_{-1}'$

In accordance with rule three, when the man is in the exhale portion of the cycle and less than 50% of volume has been exhaled, the breathhold is begun immediately.

BLOCK B-17. Set $t_{H1}' = 2t_1' + \frac{t_p'}{2}$

In the logic utilized in the program, it is implicit that the man will be in either the first or second breathing cycle when he begins to respond to the impact. Consequently, with a negative response to Block B-15, the man is in the exhale portion of the second cycle and more than 50% of the volume of air has been exhaled. Therefore, in accordance with the third breathhold rule, he will complete the exhale procedure and inhale 50% of the normal inhale volume prior to holding his breath.

BLOCK B-18. Set $j = 1$

The following blocks are concerned with determining the times at which gasps following the first breathhold time occur.

BLOCK B-19. Select at Random a Breathholding Time

From a probability distribution of breathholding times a value denoted as BHT_j is obtained. This variable, BHT_j, is expressed as an integral number of intervals of width δ.

BLOCK B-20. Store t_{Gj}'

The time at which, for example, the first gasp occurs, is equal to the time of the first breathhold plus the breathhold time. (See Figure 5.) Note that the breathhold time includes the time for exhaling.

BLOCK B-21. Is $t_{Gj}' < t_m'$?

This block inquires whether or not the j^{th} gasp occurred prior to the time

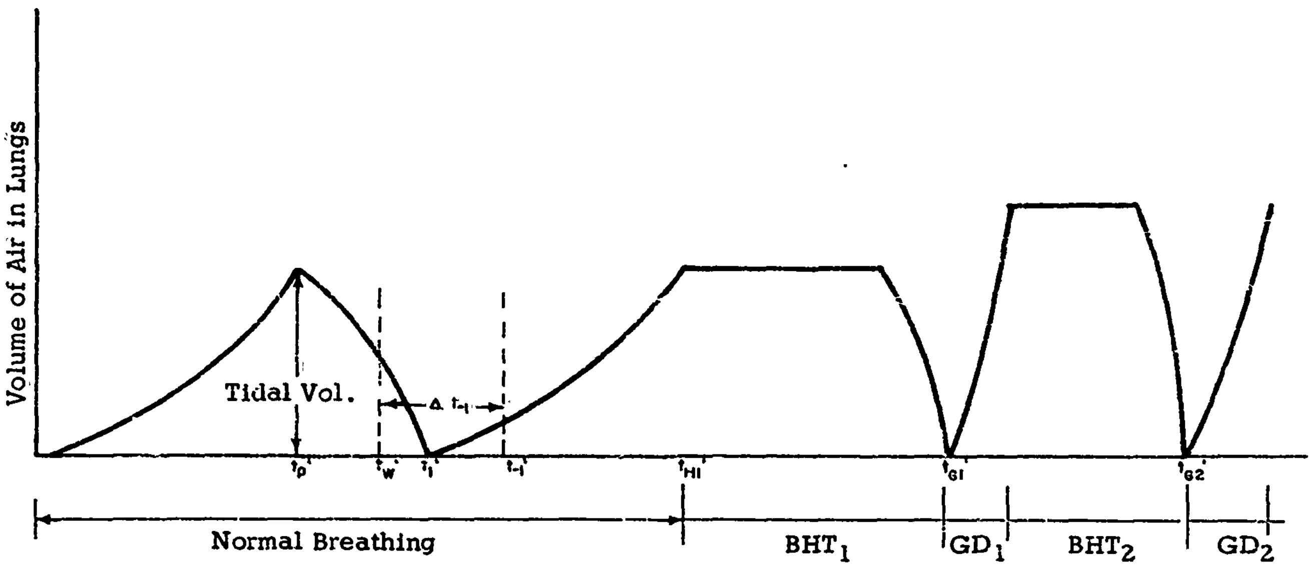


FIGURE 5. BREATHING PATTERN WHEN t_{-1} FALLS IN INHALE PORTION OF SECOND CYCLE (B13, B14 SITUATION)

at which masking was completed. If t_{Gj}' is less than t_m', then the man has not masked himself at the beginning of the j^{th} gasp. If t_{Gj}' is not less than t_m', masking has occurred prior to the beginning of the j^{th} gasp; consequently, the j^{th} gasp does not produce any vapor dose.

BLOCK B-22. Increase j by One

The test of Block B-21 is applied to the $(j + 1)^{th}$ gasp.

BLOCK B-23. Select at Random BHT_j

From a probability distribution of breathhold times, a value for the next breathhold time will be selected.

BLOCK B-24. Compute t_{Gj}'

All values of t_{Gj}' for j equal to 2, 3, ..., are obtained by adding to the time at which the previous gasp began, the gasp duration, GD, and the next breathholding time.

BLOCK B-25. Set $t_{Gj}' = \infty$

When it is determined that the masking time occurred prior to the gasping time a large number, 9.9×10^{40}, is placed in storage to represent infinity.

BLOCK B-26. Select at Random TV

The volume of air intake is selected at random from a probability distribution of tidal volumes. Note: The following breathing parameters have been generated for a specific target element:

t_{H1}'

$t_{G1}', t_{G2}', \ldots, \infty$

TV

t_m'

BLOCK B-27. Exit to Dose Subroutine

After the breathing parameters have been determined, the computer begins the dose subroutine.

Dose Routine

BLOCK R-0. Compute t_A and t_D

This block computes the times of arrival and departure of the gas cloud for a particular target from the following equations:

$$t_A = \frac{x' - 2R}{\bar{u}} \qquad \text{and} \qquad t_D = \frac{x' + 2R}{\bar{u}}$$

If the formula leads to negative t_A, t_A is set equal to zero. Both t_A and t_D are expressed as an integral number of intervals of width δ.

BLOCK R-1. Compute $t_E' = t_A + t_W$

This block computes the time of arrival of the gas cloud with respect to the beginning of the man's breathing cycle.

BLOCK R-2. Is $t_E' \geq t_m'$?

If the time at which the gas cloud arrives is greater than the time at which the man puts on his gas mask, then the man receives no dose. The effects of the next impact will then be determined with respect to this same man.

BLOCK R-3. Is $t_E' \geq t_{H1}'$?

If the time at which the gas cloud arrives is equal to or greater than the time of first breathhold, then the machine will determine the gasp dose. However, if the time at which the gas cloud arrives is less than the time of first breathhold the steady-breathing dose will be determined.

BLOCK R-4. Compute $\frac{t_{H1}' - t_E'}{\delta} = \gamma_{MAX}$

This block determines the number of increments which will be used in the numerical integration for ascertaining the quantity of vapor inhaled prior to the beginning of the first breathhold.

BLOCK R-5. Set $\gamma = 1$

$$t = t_A + \frac{\delta}{2}$$

The index, γ, is used to denote the number of the next numerical integration to be computed. The time, t, is set at the midpoint of the first interval after the gas cloud arrives.

BLOCK R-6. Determine $q = \text{MOD}\left[\frac{t_E'}{t_1'},\right] / \delta$

The breathing cycle increment number for the target, corresponding to the time at which the gas cloud arrives, is determined.

BLOCK R-7. Look-up $\Delta f(q)$

The fraction of TV inhaled in the q^{th} increment of the breathing cycle is determined from a table placed in the machine.

BLOCK R-8. Is $\Delta f(q) > 0$?

If the fraction of TV inhaled is greater than zero, the incremental volume inhaled is obtained as follows:

$$\Delta v(q) = TV\,[\Delta f\,(q)]$$

If the quantity of vapor inhaled is equal to zero, this computation is avoided.

BLOCK R-9. Is $t = t_A + (2\gamma - 1)\frac{\delta}{2} \geq t_D$?

If the mid-point of the interval is greater than the time of departure of the gas cloud, then no vapor is inhaled since the gas cloud has passed by. If the mid-point of the interval is less than the time at which the gas cloud departs, then the man may breathe while the gas cloud is around him.

BLOCK R-10. Form $[\Delta v(q)]\,[\chi(t, x', y')]$ and add to B_n

If the gas cloud has not passed by, then the quantity of vapor inhaled is computed as a product of the quantity of air inhaled times the concentration of the vapor. The concentration of the vapor is a function of the time since the impact and the x and y coordinates in terms of the wind-coordinate system. This product is then added to the previous quantity of steady-breathing dosage for the man which is stored in B_n.

BLOCK R-11. $\underline{\text{Is } \gamma = \gamma_{MAX}?}$

If γ is equal to γ_{MAX}, all of the intervals in which the man could have breathed the vapor prior to beginning the breathhold routine have been considered. The machine will next determine the quantity of vapor inhaled during the gasping routine. If γ is not equal to γ_{MAX}, (that is, γ is less than γ_{MAX}), then the subsequent interval will be considered.

BLOCK R-12. $\underline{\text{Increase } \gamma \text{ by one; (t by } \delta)}$

γ and t are incremented by one and δ, respectively, for use in determining the concentration of gas in the next interval.

BLOCK R-13. $\underline{\text{Is } t > t_m?}$

If t is greater than t_m, then the man is masked prior to the mid-point of the subsequent interval and, as a result, he will receive no vapor. The machine then will determine the effects of the next impact upon this man.

BLOCK R-14. $\underline{\text{Is } q = q_{MAX}? \quad (q_{MAX} = \frac{t_1'}{\delta})}$

If q is equal to q_{MAX}, then the dose for all intervals in this breathing cycle has been computed for this man. If q is less than q_{MAX}, then the man is still in the same breathing cycle.

BLOCK R-15. $\underline{\text{Increase q by One}}$

The index q is increased by one so that the fraction of the tidal volume inhaled during the subsequent interval can be obtained as in Block R-7.

BLOCK R-16. $\underline{\text{Set } q = 1}$

The index q is set equal to one so that the next breathing cycle for the man can be properly initiated.

BLOCK R-17. $\underline{\text{Set } j = 1}$

The index for the gasp time is set equal to one.

BLOCK R-18. $\underline{\text{Is } t_{Gj}' = \infty?}$

If t_{Gj}' is equal to ∞, this implies that the man had his gas mask on when he made the j^{th} gasp. If t_{Gj}' is not equal to ∞, the man did not have his gas mask on and the quantity of vapor inhaled will be computed later.

BLOCK R-19. Set $\gamma = 1$

The interval index γ (not the same γ as previously used) is set equal to one.

BLOCK R-20. Form $t_{Gj} = t_{Gj}' - t_w$

Set $t = t_{Gj}$

The time for the j^{th} gasp is obtained in terms of time with respect to the impact time.

BLOCK R-21. Is $t_{Gj} + GD \leq t_A$?

If t_{Gj} + GD is equal to or less than the time of arrival of the vapor cloud, then no vapor has been inhaled during the j^{th} gasp; otherwise, during the j^{th} gasp some of the gas may have been inhaled by the man.

BLOCK R-22. Increase j by One

The index j is increased by one, so that the effects during the subsequent gasp can be determined.

BLOCK R-23. Is $t_{Gj} + (2\gamma - 1)\frac{\delta}{2} \geq t_D$?

If the time of the mid-point of the γ^{th} interval during the j^{th} gasp is equal to or greater than the time of departure of the gas cloud, then no vapor was inhaled during the γ^{th} interval, otherwise some vapor may have been inhaled during the γ^{th} interval.

BLOCK R-24. Is $t_{Gj} + (2\gamma - 1)\frac{\delta}{2} > t_m$?

If the time of the mid-point of the γ^{th} interval on the j^{th} gasp is greater than the time of masking, then no vapor was inhaled during the γ^{th} interval. The effects of the next impact are then ascertained, otherwise, some vapor may have been inhaled during the γ^{th} interval.

BLOCK R-25. Form $\Delta v_{g\gamma} \chi(t_{Gj} + (2\gamma - 1)\frac{\delta}{2})$ and add to (G_n)

Vapor dose in the γ^{th} interval of the j^{th} gasp is obtained by obtaining the product of the volume inhaled and the vapor concentration.

BLOCK R-26. Is $\gamma = \frac{GD}{\delta}$?

If γ is equal to GD divided by δ, then the complete vapor dose due to the j^{th} gasp has been ascertained and the subsequent gasp will be considered.

BLOCK R-27. Increase γ by One, t by δ

The index γ and time t are incremented by one and δ, respectively, in order to evaluate the vapor dose during the next interval.

IV. COMPUTER PROGRAM

4.1 This section contains the literal computer programs in ALGOL and FORTRAN. These are the specific forms in which the simulation is read into the computer.

4.2 At present a detailed annotation of the programs relating them to the blocks in the flow charts has not been made. This will be carried out in the next contract year.

SIMULATION MODEL OF THE
155-mm HOWITZER WEAPON SYSTEM

THE ALGOL VERSION OF THE COMPUTER SIMULATION

```
    BURROUGHS ALGEBRAIC COMPILER - STANDARD VERSION   8/10/61
INTEGER I... , J... , K... , L... , M... , N...                         $
ARRAY    XMAN(20) , YMAN(20) , XMANW(20) , YMANW(20) , IK(3)            ,
         XIMP(18) , YIMP(18) , XIMPW(18) , YIMPW(18) , D(4,20)          ,
         XX(3) , YY(3) , XS(3) , YS(3) , TGP(20) , DF(50) , DVG(50) ,
         AMBDA(10) , A(10) , B(10) , C(10) , S(10) , ALPHA(10,2)        ,
         DOS(10,10) , IXIMX(10) , RSQR(10,10) , CD(4,20)                $
PROCEDURE    RN( L $ K , M , N , X )                                    $  RN    01
BEGIN    INTEGER  I... , J... , K... , L... , M... , N...               $  RN    02
         L = 8193L + 1                                                  $  RN    03
         K = MOD( L , 131072 )                                          $  RN    04
         X = FLOAT(K)/131072.0                                          $  RN    05
         M = 40.0X                                                      $  RN    06
         Y = ENTIRE(1000.0X)                                            $  RN    07
         N = Y - 25M                                                    $  RN    08
         RETURN                                   END         RN()      $  RN    09
PROCEDURE    RANDOM(  I $$RN() )                                        $  RANDM00
BEGIN    INTEGER  I... , J... , K... , L... , M... , N...               $  RANDM01
         ARRAY    A(12,42)                                              $  RANDM02
         RN( K $ L , M , N , X )                                        $  RANDM03
         K = L                                                          $  RANDM04
         IF    I GTR 15                                                 $  RANDM05
```

```
BEGIN   K = I                                                      $ RANDM06
        J = I                                                      $ RANDM07
        GO  TO  R5                                      END        $ RANDM08
        SWITCH  I , (R1,R1,R1,R1,R1,R1,R1,R1,R1,R1,R1,R1,R2,R3,R4) $ RANDM09
        READ($$DIS)                                                $ RANDM10
        GO  TO  R5                                                 $ RANDM11
R1..    EITHER  IF  RAN3 LEQ 0.0                                   $ RANDM12
        RA = A(I,42)                                               $ RANDM13
        OTHERWISE                                                  $ RANDM14
        RA = A(I,M+1) + (FLOAT(N)/25.0)(A(I,M+2) - A(I,M+1))       $ RANDM15
        GO  TO  R5                                                 $ RANDM16
R2..    EITHER  IF  RAN3 LEQ 0.0                                   $ RANDM17
        RA = RAN1                                                  $ RANDM18
        OTHERWISE                                                  $ RANDM19
        RA = X                                                     $ RANDM20
        GO  TO  R5                                                 $ RANDM21
R3..    EITHER  IF  RAN3 LEQ 0.0                                   $ RANDM22
        RA = RAN2                                                  $ RANDM23
        OTHERWISE                                                  $ RANDM24
        RA = 1000.0X                                               $ RANDM25
        GO  TO  R5                                                 $ RANDM26
R4..    WRITE($$FMTH)                                              $ RANDM27
```

```
         FOR   N = ( 1 , 1 , 42 )                                                $  RANDM28
BEGIN    V = 0.025FLOAT(N-1)                                                     $  RANDM29
         WRITE($$OUT1,FMT1)                                    END               $  RANDM30
         WRITE($$OUT2,FMT2)                                                      $  RANDM31
         GO  TO  R6                                                              $  RANDM32
R5..     RANDOM() = RA                                                           $  RANDM33
INPUT    DIS(RAN1,RAN2,RAN3,FOR L=(1,1,12) $ FOR I=(1,1,42) $ A(L,I) ) $  RANDM34
OUTPUT   OUT1( V , FOR M=(1,1,12)  $  A(M,N) )                                   ,  RANDM35
         OUT2( J , K )                                                           $  RANDM36
FORMAT   FMTH(W3,(W4,(B46,*PROBABILITY  DISTRIBUTIONS*,W4,(*    X    *,B6,  RANDM37
             *1         2         3         4         5         6 *,B7,  RANDM38
             *7         8         9        10        11        12*,W4)))) ,  RANDM39
         FMT1(X7.3,5X10.3,7X9.3,W0)                                              ,  RANDM40
         FMT2(W0,(B30,*INITIAL  RANDOM  NUMBER  WAS*,I8                          ,  RANDM41
             *    LAST  RANDOM  NUMBER  USED  WAS*,I8,W4))                        $  RANDM42
R6..     RETURN                                                END    RANDOM()  $  RANDM43
         SUBROUTINE   TARGT                                                      $  TARGT01
BEGIN    L = 1000/NMAX                                                           $  TARGT02
         GO  TO  T1                                                              $  TARGT03
         UNTIL   I NEQ NMAX                                                      $  TARGT04
T1..     I = FIX( RANDOM( 14 $$RN() ) )/L                                        $  TARGT05
         GO  TO  T2                                                              $  TARGT06
```

```
        UNTIL   ( . NEQ NMAX ) AND ( I NEQ J )                      $ TARGT07
T2..    J = FIX( RANDOM( 14 $$RN() ) )/L                            $ TARGT08
        GO  TO  T3                                                  $ TARGT09
        UNTIL   ( K NEQ NMAX ) AND ( K NEQ J ) AND ( K NEQ I )      $ TARGT10
T3..    K = FIX( RANDOM( 14 $$RN() ) )/L                            $ TARGT11
        IK(1) = I + 1                                               $ TARGT12
        IK(2) = J + 1                                               $ TARGT13
        IK(3) = K + 1                                               $ TARGT14
        FOR   L = ( 1 , 1 , 3 )                                     $ TARGT15
BEGIN   XX(L) = XMAN(IK(L))                                         $ TARGT16
        YY(L) = YMAN(IK(L))                                         $ TARGT17
        XS(L) = XX(L) + RANDOM(  1 $$RN() )                         $ TARGT18
        YS(L) = YY(L) + RANDOM(  1 $$RN() )           END           $ TARGT19
        R3 = ( XS(1) + XS(2) + XS(3) )/3.0                          $ TARGT20
        R4 = ( YS(1) + YS(2) + YS(3) )/3.0                          $ TARGT21
        R5 = R3 + RANDOM(  2 $$RN() )                               $ TARGT22
        R6 = R4 + RANDOM(  3 $$RN() )                               $ TARGT23
        X = FLOAT( IMAX + 1 )ZETA/2.0                               $ TARGT24
        FOR   L = ( 1 , 1 , IMAX )                                  $ TARGT25
BEGIN   XIMP(L) = RANDOM(  4 $$RN() ) + R5 - X + L.ZETA             $ TARGT26
        YIMP(L) = RANDOM(  5 $$RN() ) + R6                          $ TARGT27
        XIMPW(L) = XIMP(L)COST + YIMP(L)SINT                        $ TARGT28
```

```
          YIMPW(L) = YIMP(L)COST - XIMP(L)SINT      END              $ TARGT29
          R1 = ( XX(1) + XX(2) + XX(3) )/3.0                         $ TARGT30
          R2 = ( YY(1) + YY(2) + YY(3) )/3.0                         $ TARGT31
          RETURN                                    END     TARGT    $ TARGT32
          SUBROUTINE   FRAGM                                         $ FRAGM01
BEGIN     FOR  N  =  ( 1 , 1 , NMAX )                                $ FRAGM02
BEGIN     FOR  I  =  ( 1 , 1 , IMAX )                                $ FRAGM03
BEGIN     IF   YIMP(I)  GTR  YMAN(N)                                 $ FRAGM04
          GO  TO  F5                                                 $ FRAGM05
          X  =  ( XMAN(N) - XIMP(I) )                                $ FRAGM06
          Y  =  ( YMAN(N) - YIMP(I) )                                $ FRAGM07
          DSQR  =  X.X + Y.Y                                         $ FRAGM08
          IF   DSQR  GTR  RSM                                        $ FRAGM09
          GO  TO  F5                                                 $ FRAGM10
          IF   DSQR  LEQ  DELTA                                      $ FRAGM11
          GO  TO  F3                                                 $ FRAGM12
          EITHER  IF   Y EQL 0                                       $ FRAGM13
          BETA  =  9.9**40                                           $ FRAGM14
          OTHERWISE                                                  $ FRAGM15
          BETA  =  ABS( X/Y )                                        $ FRAGM16
          ITAU  =  1                                                 $ FRAGM17
F1..      IXI  =  1                                                  $ FRAGM18
```

```
         IF   ALPHA(ITAU,1)  GTR  BETA                          $ FRAGM19
         GO  TO  F5                                             $ FRAGM20
         IF   ALPHA(ITAU,2)  LSS  BETA                          $ FRAGM21
         GO  TO  F4                                             $ FRAGM22
F2..     IF   DSQR  LEQ  RSQR(ITAU,IXI)                         $ FRAGM23
BEGIN    PROB  =  RANDOM( 13 **RN() )                           $ FRAGM24
         IF   PROB  LEQ  DOS(ITAU,IXI)                          $ FRAGM25
F3..     D(4,N)  =  D(4,N) + 1                                  $ FRAGM26
         GO  TO  F5                           END               $ FRAGM27
         IF   IXI  LSS  IXIMX(ITAU)                             $ FRAGM28
BEGIN    IXI  =  IXI + 1                                        $ FRAGM29
         GO  TO  F2                           END               $ FRAGM30
         GO  TO  F5                                             $ FRAGM31
F4..     IF   ITAU  EQL  ITMAX                                  $ FRAGM32
         GO  TO  F5                                             $ FRAGM33
         ITAU  =  ITAU  + 1                                     $ FRAGM34
         GO  TO  F1                                             $ FRAGM35
F5..END                                       END               $ FRAGM36
         RETURN                               END       FRAGM   $ FRAGM37
         SUBROUTINE   SPLSH                                     $ SPLSH01
BEGIN    FOR   N = ( 1 , 1 , NMAX )                             $ SPLSH02
         FOR   I = ( 1 , 1 , IMAX )                             $ SPLSH03
```

```
BEGIN   K = KMAX                                                $ SPLSH04
        R0 = YMAN(N) - YIMP(I)                                  $ SPLSH05
        R1 = R0 - AMBDA(K)                                      $ SPLSH06
        R2 = XMAN(N) - XIMP(I)                                  $ SPLSH07
        IF  ( ABS(R1) GTR A(K) ) OR ( ABS(R2) GTR B(K) )        $ SPLSH08
        GO  TO  S3                                              $ SPLSH09
        R2 = R2.R2                                              $ SPLSH10
        R5 = R1 + C(K)                                          $ SPLSH11
        R6 = R1 - C(K)                                          $ SPLSH12
        R3 = R2 + R5.R5                                         $ SPLSH13
        R4 = R2 + R6.R6                                         $ SPLSH14
        RHO = SQRT(R3) + SQRT(R4)                               $ SPLSH15
        IF  RHO GTR 2A(K)                                       $ SPLSH16
        GO  TO  S3                                              $ SPLSH17
S1..    IF  K EQL 1                                             $ SPLSH18
        GO  TO  S2                                              $ SPLSH19
        K = K - 1                                               $ SPLSH20
        R1 = R0 - AMBDA(K)                                      $ SPLSH21
        R5 = R1 + C(K)                                          $ SPLSH22
        R6 = R1 - C(K)                                          $ SPLSH23
        R3 = R2 + R5.R5                                         $ SPLSH24
        R4 = R2 + R6.R6                                         $ SPLSH25
```

```
         RHO = SQRT(R3) + SQRT(R4)                                    $ SPLSH26
         IF   RHO LEQ 2A(K)                                           $ SPLSH27
         GO  TO  S1                                                   $ SPLSH28
         K = K + 1                                                    $ SPLSH29
S2..     D(1,N) = D(1,N) + S(K)                                       $ SPLSH30
S3..END                                                               $ SPLSH31
        RETURN                                     END        SPLSH   $ SPLSH32
         SUBROUTINE   VAPOR                                           $ VAPOR01
BEGIN    FOR   N = ( 1 , 1 , NMAX )                                   $ VAPOR02
BEGIN    BPS = 1.0                                                    $ VAPOR03
         FOR   I = ( 1 , 1 , IMAX )                                   $ VAPOR04
BEGIN    X = XMANW(N) - XIMPW(I)                                      $ VAPOR05
         Y = YMANW(N) - YIMPW(I)                                      $ VAPOR06
         IF  NOT ((ABS(Y) LEQ 2R) AND (X GTR -R) AND (X LSS 1000.0))  $ VAPOR07
         GO  TO  V1                                                   $ VAPOR08
         IF   BPS EQL 1.0                                             $ VAPOR09
COMMENT              BREATHING   PARAMETER   SUBROUTINE               $ BPARA00
BEGIN    BPS  =  0.0                                                  $ BPARA01
         TW = DELT.ENTIRE((T1P.RANDOM( 13 $$RN() ) + DELT2)/DELT)     $ BPARA02
         IF   TW  EQL  T1P                                            $ BPARA03
         TW = 0                                                       $ BPARA04
         TM1 = DELT.ENTIRE((RANDOM(  6 $$RN() ) + DELT2)/DELT)        $ BPARA05
```

```
         TM1P   =   TM1 + TW                                          $  BPARA06
         TM  = DELT.ENTIRE((RANDOM(  7 $$RN()  ) + DELT2)/DELT) + TM1  $  BPARA07
         TMP   =   TM + TW                                            $  BPARA08
         EITHER  IF    TM1P   LEQ   TPP                               $  BPARA09
         TH1P   =   TPP                                               $  BPARA10
         OR      IF   2TM1P   LEQ   TPP + T1P                         $  BPARA11
         TH1P   =   TM1P                                              $  BPARA12
         OR      IF    TM1P   LEQ   T1P                               $  BPARA13
         TH1P   =   T1P + TPP/2.0                                     $  BPARA14
         OR      IF    TM1P   LEQ   T1P + TPP                         $  BPARA15
         TH1P   =   T1P + TPP                                         $  BPARA16
         OR      IF   2TM1P   LEQ   3T1P + TPP                        $  BPARA17
         TH1P   =   TM1P                                              $  BPARA18
         OTHERWISE                                                    $  BPARA19
         TH1P   =   2T1P + TPP/2.0                                    $  BPARA20
         J  =  1                                                      $  BPARA21
         BHT = DELT.ENTIRE((RANDOM(  8 $$RN()  ) + DELT2)/DELT)        $  BPARA22
         TGP(J)   =   TH1P + BHT                                      $  BPARA23
         GO  TO  B2                                                   $  BPARA24
B1..     J  =  J + 1                                                  $  BPARA25
         BHT = DELT.ENTIRE((RANDOM(  8 $$RN()  ) + DELT2)/DELT)        $  BPARA26
         TGP(J)   =   TGP(J-1) + GD + BHT                             $  BPARA27
```

```
B2..    IF   TGP(J)  LSS  TMP                                    $  BPARA28
        GO  TO  B1                                               $  BPARA29
        TGP(J) = 9.9**40                                         $  BPARA30
        TV = RANDOM(  9 $$RN() )                     END         $  BPARA31
        EITHER  IF   X  LEQ  R                                   $  VAPOR10
        TA = 0                                                   $  VAPOR11
        OTHERWISE                                                $  VAPOR12
        TA = DELT.ENTIRE((( X - R )/UBAR + DELT2)/DELT)          $  VAPOR13
        TD = DELT.ENTIRE((( X + R )/UBAR + DELT2)/DELT)          $  VAPOR14
COMMENT                         DOSE   SUBROUTINE                $  DOSE 01
BEGIN   TEP = TA + TW                                            $  DOSE 02
        IF   TEP GEQ TMP                                         $  DOSE 03
        GO  TO  V1                                               $  DOSE 04
        IF   TEP GEQ THIP                                        $  DOSE 05
        GO  TO  E3                                               $  DOSE 06
        MGMAX = ( THIP - TEP )/DELT                              $  DOSE 07
        K = 1                                                    $  DOSF 08
        T = TA + DELT2                                           $  COSE 09
        MQ = MOD(FIX(TEP/DELT),MQMAX) + 1                        $  DODE 10
E1..    IF   DF(MQ) LEQ 0                                        $  DOSE 11
        GO  TO  E2                                               $  DOSE 12
        IF   T GEQ TD                                            $  DOSE 13
```

```
         GO  TO  V1                                 $ DOSE 14
         ENTER    CHIF                              $ DOSE 15
         D(2,N) = D(2,N) + DF(MQ)CHI.TV             $ DOSE 16
E2..     IF   K EQL MGMAX                           $ DOSE 17
         GO  TO  E3                                 $ DOSE 18
         K = K + 1                                  $ DOSE 19
         T = T + DELT                               $ DOSE 20
         IF   T GTR TM                              $ DOSE 21
         GO  TO  V1                                 $ DOSE 22
         EITHER   IF   MQ EQL MQMAX                 $ DOSE 23
         MQ = 1                                     $ DOSE 24
         OTHERWISE                                  $ DOSE 25
         MQ = MQ + 1                                $ DOSE 26
         GO  TO  F1                                 $ DOSE 27
F3..     J = 0                                      $ DOSE 28
E4..     J = J + 1                                  $ DOSE 29
         IF   TGP(J) EQL 9.9**40                    $ DOSE 30
         GO  TO  V1                                 $ DOSE 31
         T = TGP(J) - TW                            $ DOSE 32
         IF   T + GD LEQ TA                         $ DOSE 33
         GO  TO  E4                                 $ DOSE 34
         T = T - DELT2                              $ DOSE 35
```

```
         FOR    K = ( 1 , 1 , MKMAX )                                   $  DOSE 36
BEGIN    T = T + DELT                                                   $  DOSE 37
         IF    ( T GEQ TD ) OR ( T GTR TM )                             $  DOSE 38
         GO  TO  V1                                                     $  DOSE 39
         ENTER    CHIF                                                  $  DOSE 40
         D(3,N) = D(3,N) + DVG(K)CHI                  END               $  DOSE 41
         GO  TO  E4                                   END               $  DOSE 42
V1..END                                               END               $  VAPOR15
         RETURN                                       END       VAPOR   $  VAPOR16
         SUBROUTINE    CHIF                                             $  CHIF 01
BEGIN    IF    T GTR 120.0                                              $  CHIF 02
         GO  TO  V1                                                     $  CHIF 03
         R1 = UBAR.T                                                    $  CHIF 04
         R2 = X - R1                                                    $  CHIF 05
         R3 = R2.R2 + Y.Y                                               $  CHIF 06
         R2 = ( ( R1 + DR )/100.0 )*ALPA                                $  CHIF 07
         R4 = R2.R2                                                     $  CHIF 08
         CHI = ( Q.EXP( -R3/(23.2562R4) ) )/( 402.90378R4.R2 )          $  CHIF 09
         RETURN                                       END       CHIF    $  CHIF 10
         RANDOM(  0 $$RN() )                                            $
START..  READ($$GEN)                                                    $
         IF    M NEQ 0                                                  $
```

```
        RANDOM(  M $$RN() )                                        $
        PI = 3.1415927                                             $
        R1 = 0.01745323?                                           $
        THETA = R1.THETA                                           $
        COST = COS(THETA)                                          $
        SINT = SIN(THETA)                                          $
        DELT2 = DELT/2.0                                           $
        MKMAX = GD/DELT                                            $
        MOMAX = TIP/DELT                                           $
        ALPA = ALPA + RANDOM( 11 $$RN() )                          $
        DR = 100( R/10.23 )*(1.0/ALPA)                             $
        UBAR = UBAR + RANDOM( 12 $$RN() )                          $
        Q = RANDOM( 10 $$RN() )                + 3000000.0         $
        FOR   X = TIP , TPP , GD                                   $
        IF   ( X - DELT.ENTIRE(X/DELT) )  NEQ  0                   $
        STOP    9769669669                                         $
        READ($$DAT)                                                $
        FOR   I = ( 1 , 1 , NMAX )                                 $
BEGIN   XMANW(I) = XMAN(I)COST + YMAN(I)SINT                       $
        YMANW(I) = YMAN(I)COST - XMAN(I)SINT          END          $
        FOR   I = 1 , 2                                            $
        FOR   J = ( 1 , 1 , ITMAX )                                $
```

```
        EITHER   IF   ALPHA(J,I)  EQL  90.0                      $
        ALPHA(J,I) = 0                                           $
        OTHERWISE                                                $
        ALPHA(J,I) = 1.0/TAN(R1.ALPHA(J,I))                      $
        FOR   I = ( 1 , 1 , 4 )                                  $
        FOR   J = ( 1 , 1 , 20 )                                 $
        CD(I,J) = 0                                              $
        FOR   IREP = ( 1 , 1 , IREPM )                           $
BEGIN   FOR   I = ( 1 , 1 , 4 )                                  $
        FOR   J = ( 1 , 1 , 20 )                                 $
        D(I,J) = 0                                               $
        ENTER    TARGT                                           $
        IF   FRA EQL 1.0                                         $
        ENTER    FRAGM                                           $
        IF   SPL EQL 1.0                                         $
        ENTER    SPLSH                                           $
        IF   VAP EQL 1.0                                         $
        ENTER    VAPOR                                           $
        IF   BDS EQL 1.0                                         $
BEGIN   WRITE($$OUT4,FMT4)                                       $
        WRITE($$OUT7,FMT6)                          END          $
        FOR   I = ( 1 , 1 , 4 )                                  $
```

```
         FOR   J = ( 1 , 1 , 20 )                                   $
         CD(I,J) = CD(I,J)  +  D(I,J)                               $
         IREP = IREP                               END              $
         WRITE($$OUT5,FMT5)                                         $
         WRITE($$OUT6,FMT6)                                         $
         RANDOM( 15 $$RN() )                                        $
         GO  TO  START                                              $
INPUT    GEN( RSM , DELTA , TPP , TIP , DELT , R , UBAR , ALPA , THETA ,
         GD , ZETA , FRA , SPL , VAP , BDS                          ,
         ITMAX , IMAX , KMAX , NMAX , IREPM , M )                   ,
         DAT((FOR K=(1,1,KMAX) $ (AMBDA(K),A(K),B(K),C(K),S(K)))    ,
             (FOR K=(1,1,NMAX) $(XMAN(K),YMAN(K)))                  ,
             (FOR K=(1,1,MQMAX) $ DF(K))                            ,
             (FOR K=(1,1,MQMAX) $ DVG(K))                           ,
             (FOR K=(1,1,ITMAX) $ (IXIMX(K),ALPHA(K,1),ALPHA(K,2)   ,
              FOR I=(1,1,IXIMX(K)) $ (RSQR(K,I),DOS(K,I))))))       $
OUTPUT   OUT4(IREP)                                                 ,
         OUT5(IREPM)                                                ,
         OUT6(FOR N=(1,1,NMAX) $ ( N , XMAN(N) , YMAN(N) , XMANW(N) ,
              YMANW(N) , FOR I=(1,1,4) $CD(I,N) ) )                 ,
         OUT7(FOR N=(1,1,NMAX) $ ( N , XMAN(N) , YMAN(N) , XMANW(N) ,
              YMANW(N) , FOR I=(1,1,4) $ D(I,N) ) )                 $
```

```
FORMAT  FMT4(B48,*DOSSAGE  FOR  VOLLEY*,I4,W3,(W4))                    ,
        FMT5(B41,*COMMULATIVE  DOSSAGE  FOR*,I4,*  VOLLEYS*,W3,(W4))   ,
        FMT6(*   N   XMAN(N)   YMAN(N)  XMANW(N)  YMANW(N)*,B8         ,
            *SPLASH           BREATHE              GASP          FRAGM*,
            *ENTATION*,W4,(W4,(I4,4X10.2,4F18.8,W4)))                  $
        FINISH                                                         $
      COMPILED PROGRAM ENDS AT 2477
      NEXT AVAILABLE CELL IS 4668
```

PROBABILITY DISTRIBUTIONS

X	1	2	3	4	5	6	7	8	9	10	11	12
.000	-129.071	-149.232	-274.002	-57.452	-112.936	.100	6.500	2.000	.600	.000	.000	.000
.025	-93.924	-107.554	-197.640	-40.706	-81.396	.131	9.188	3.125	.636	.000	.000	.000
.050	-78.194	-90.407	-166.163	-34.212	-68.420	.163	9.500	3.750	.671	.000	.000	.000
.075	-68.320	-78.390	-145.134	-29.896	-59.780	.194	9.708	4.250	.707	.000	.000	.000
.100	-60.813	-70.310	-129.228	-26.611	-53.212	.207	9.917	4.750	.743	.000	.000	.000
.125	-54.594	-63.120	-116.013	-23.886	-47.770	.216	10.150	5.375	.779	.000	.000	.000
.150	-47.224	-56.911	-104.493	-21.520	-43.071	.225	10.400	6.000	.813	.000	.000	.000
.175	-44.357	-51.282	-94.248	-19.412	-38.812	.234	10.625	6.505	.844	.000	.000	.000
.200	-39.959	-46.198	-84.911	-17.435	-34.964	.243	10.833	7.000	.875	.000	.000	.000
.225	-35.851	-41.447	-76.185	-15.691	-31.369	.252	11.050	7.583	.905	.000	.000	.000
.250	-32.020	-37.018	-68.032	-14.014	-28.017	.261	11.300	8.000	.930	.000	.000	.000
.275	-28.371	-32.790	-60.288	-12.417	-24.825	.270	11.500	8.417	.955	.000	.000	.000
.300	-24.879	-28.761	-52.861	-10.889	-21.769	.279	11.800	8.833	.980	.000	.000	.000
.325	-21.540	-24.901	-45.773	-9.426	-18.848	.288	12.050	9.250	1.004	.000	.000	.000
.350	-18.288	-21.140	-38.555	-8.003	-16.002	.296	12.300	9.666	1.023	.000	.000	.000
.375	-15.145	-17.505	-32.173	-6.624	-13.251	.305	12.550	10.083	1.042	.000	.000	.000
.400	-12.019	-13.892	-25.539	-5.260	-10.516	.313	12.800	10.500	1.062	.000	.000	.000
.425	-8.951	-10.344	-19.008	-3.919	-7.832	.321	13.042	10.917	1.081	.000	.000	.000
.450	-5.954	-6.870	-12.653	-2.606	-5.210	.329	13.250	11.333	1.100	.000	.000	.000
.475	-2.972	-3.452	-6.323	-1.302	-2.601	.337	13.450	11.875	1.119	.000	.000	.000
.500	.000	.000	.000	.000	.000	.345	13.700	12.500	1.138	.000	.000	.000
.525	2.972	3.452	6.323	1.302	2.601	.353	13.950	13.125	1.158	.000	.000	.000
.550	5.954	6.870	12.653	2.606	5.210	.361	14.200	13.750	1.177	.000	.000	.000
.575	8.951	10.345	19.008	3.919	7.332	.369	14.450	14.168	1.196	.000	.000	.000
.600	12.019	13.892	25.539	5.260	10.516	.377	14.700	14.500	1.217	.000	.000	.000
.625	15.144	17.505	32.173	6.624	13.251	.385	14.950	15.125	1.233	.000	.000	.000
.650	18.288	21.140	38.555	8.003	16.002	.394	15.200	15.750	1.258	.000	.000	.000
.675	21.540	24.905	45.773	9.426	18.848	.402	15.450	16.375	1.279	.000	.000	.000
.700	24.879	28.761	52.861	10.889	21.769	.410	15.700	17.000	1.300	.000	.000	.000
.725	28.371	32.790	60.288	12.417	24.825	.418	16.062	17.583	1.328	.000	.000	.000
.750	32.021	37.015	68.032	14.034	28.017	.427	16.375	18.000	1.356	.000	.000	.000
.775	35.851	41.447	76.185	15.691	31.369	.435	16.875	18.750	1.383	.000	.000	.000
.800	39.951	46.198	84.911	17.435	34.964	.443	17.250	19.500	1.413	.000	.000	.000
.825	44.357	51.297	94.248	19.412	38.812	.452	17.625	20.125	1.450	.000	.000	.000
.850	47.224	56.911	104.493	21.520	43.071	.460	18.500	20.750	1.486	.000	.000	.000
.875	54.594	63.120	116.013	23.886	47.770	.468	19.250	21.375	1.538	.000	.000	.000
.900	60.813	70.310	129.228	26.611	53.212	.477	20.500	22.000	1.600	.000	.000	.000
.925	68.320	78.390	145.134	29.896	59.780	.485	21.750	22.875	1.667	.000	.000	.000
.950	78.194	90.407	166.163	34.212	68.420	.493	22.500	24.000	1.750	.000	.000	.000
.975	93.924	107.554	197.640	40.706	81.396	.517	23.750	25.250	1.950	.000	.000	.000
1.000	129.071	149.232	274.002	57.452	112.936	.600	25.000	26.500	2.330	.000	.000	.000
1.025	40.000	-30.000	50.000	-15.010	30.000	.300	15.000	12.000	1.500	.000	.000	.000

INITIAL RANDOM NUMBER WAS 5671 LAST RANDOM NUMBER USED WAS 104191

SIMULATION MODEL OF THE
155-mm HOWITZER WEAPON SYSTEM

THE FORTRAN VERSION OF THE COMPUTER SIMULATION

```
*3076000 01 003 7222039999990001 1020 GRAVES      GRAVES
*    XEQ
*    LIBE
*    FORTRAN
     CALLTAPE(2,IN,IO)
     DIMENSION XMAN(100) , YMAN(100) , XMANW(100) , YMANW(100) ,      DRIVR
    1   IK(3) , XIMP(18) , YIMP(18) , XIMPW(18) , YIMPW(18) ,        DRIVR
    2   D(5,100) , XX(3) , YY(3) , XS(3) , YS(3) , TGP(20) ,         DRIVR
    3   DF(50) , DVG(50) , AMBDA(10) , A(10) , B(10) , C(10) ,       DRIVR
    4   S(10) , ALPHA(10,2) , DOS(10,10) , IXIMX(10) , RSQR(10,10)   DRIVR
     X = RANDOM(0)                                                    DRIVR
   5 READ  INPUT  TAPE  IN ,  100 , RSM , DELTA , TFP , T1P , DELT ,  DRIVR
    1                           R , UBAR , ALPA , THETA , GD ,        DRIVR
    2                           ZETA , FRA , SPL , VAP , BDS          DRIVR
     READ  INPUT  TAPE  IN ,  101 , ITMAX , IMAX , KMAX , NMAX ,      DRIVR
    1                           IREPM , M                             DRIVR
     IF  (M)                 10 ,  15 ,  10                           DRIVR
  10 X = RANDOM(M)                                                    DRIVR
  15 PI = 3.1415927                                                   DRIVR
     R1 = 0.017453293                                                 DRIVR
     THETA = R1*THETA                                                 DRIVR
     COST = COSF(THETA)                                               DRIVR
```

```
      SINT = SINF(THETA)                                               DRIVR
      DELT2 = DELT/2.0                                                 DRIVR
      MKMAX = GD/DELT                                                  DRIVR
      MQMAX = T1P/DELT                                                 DRIVR
      ALPA = ALPA + RANDOM(11)                                         DRIVR
      DR = 1000.0*((R/10.23)**(1.C/ALPA))                              DRIVR
      UBAR = UBAR + RANDOM(12)                                         DRIVR
      Q = RANDOM(10)                                  +3000000.0       DRIVR
      IF    ( T1P - DELT*INTF(T1P/DELT) )        99 ,   20 ,   99      DRIVR
   20 IF    ( TPP - DELT*INTF(TPP/DELT) )        99 ,   25 ,   99      DRIVR
   25 IF    (  GD - DELT*INTF( GD/DELT) )        99 ,   30 ,   99      DRIVR
   30 READ  INPUT  TAPE  IN ,  102 , ( AMBDA(K) , A(K) , B(K) , C(K) , DRIVR
     1                        S(K) , K = 1 , KMAX )                    DRIVR
      READ  INPUT  TAPE  IN ,  103 , ( XMAN(N),YMAN(N), N = 1 , NMAX ) DRIVR
      READ  INPUT  TAPE  IN ,  104 , ( DF(K) , K = 1 , MQMAX )         DRIVR
      READ  INPUT  TAPE  IN ,  104 , ( DVG(K) , K = 1 , MQMAX )        DRIVR
      READ  INPUT  TAPE  IN ,  105 , ( IXIMX(K),ALPHA(K,1),ALPHA(K,2), DRIVR
     1                        ( RSQR(K,I),DOS(K,I),I=1,5),             DRIVR
     2                        K=1,ITMAX )                              DRIVR
      DO     35   I = 1 , NMAX                                         DRIVR
      XMANW(I) = XMAN(I)*COST + YMAN(I)*SINT                           DRIVR
   35 YMANW(I) = YMAN(I)*COST - XMAN(I)*SINT                           DRIVR
```

```
      DO     40   I = 1 , 2                                          DRIVR
      DO     40   J = 1 , ITMAX                                      DRIVR
      IF   ( ALPHA(J,I) - 90.0 )      39  ,  38  ,  39               DRIVR
   38 ALPHA(J,I) = 0.0                                               DRIVR
      GO  TO  40                                                     DRIVR
   39 ALPHA(J,I) = 1.0/TANF(R1*ALPHA(J,I))                           DRIVR
   40 CONTINUE                                                       DRIVR
      DO     98   IREP = 1 , IREPM                                   DRIVR
      DO     47   I = 1 , 4                                          DRIVR
      DO     47   J = 1 , 20                                         DRIVR
   47 D(I,J) = 0.0                                                   DRIVR
   50 GO  TO  2002                                                   DRIVR
   52 IF   ( FRA - 1.0 )          54 , 3002 ,   54                   DRIVR
   54 IF   ( SPL - 1.0 )          56 , 4002 ,   56                   DRIVR
   56 IF   ( VAP - 1.0 )          58 , 5002 ,   58                   DRIVR
   58 CONTINUE
      WRITE  OUTPUT  TAPE  IO , 107 , IREP
   98 WRITE  OUTPUT  TAPE  IO ,  106 , ( N , XMAN(N) , YMAN(N) ,
     1                               XMANW(N) , YMANW(N) , ( D(I,N) ,
     2                               I=1,4 ) , N=1,NMAX )
       X = RANDOM(15)
   99 CALL   DUMP                                                    DRIVR
```

```
  100 FORMAT(5F10.1)                                                        DRIVR
  101 FORMAT(6I10)                                                          DRIVR
  102 FORMAT(5F10.1)                                                        DRIVR
  103 FORMAT(2F20.1)                                                        DRIVR
  104 FORMAT(10F7.2)                                                        DRIVR
  105 FORMAT(I3,4F5.1,F6.2,F5.2,F[illegible].2,F5.2,2F6.2,F8.1,F6.1)
  106 FORMAT(I4,4F10.2,4E18.8)
  107 FORMAT(1H0,47(1H ),22HDOSSAGE  FOR  VOLLEY   ,I4//
     1     40H   N   XMAN(N)   YMAN(N)  XMANW(N)  YMAN    ,
     2     40HW(N)  YMANW(N)       SPLASH           B    ,
     3     40HREATHE       GASP        FRAGMENTATION       ///   )
C2001      SELECTION  OF  TARGETS  AND  DELIVERY  OF  VOLLEY                TARGT
 2002 L = 1000/NMAX                                                         TARGT
 2003 I = XFIXF(RANDOM(14))/L                                               TARGT
 2004 IF   ( NMAX - I )   2003 , 2003 , 2005                                TARGT
 2005 J = XFIXF(RANDOM(14))/L                                               TARGT
 2006 IF   ( NMAX - J )   2005 , 2005 , 2007                                TARGT
 2007 IF   ( I - J )      2008 , 2005 , 2008                                TARGT
 2008 K = XFIXF(RANDOM(14))/L                                               TARGT
 2009 IF   ( NMAX - K )   2008 , 2008 , 2010                                TARGT
 2010 IF   ( I - K )      2011 , 2008 , 2011                                TARGT
 2011 IF   ( J - K )      2012 , 2008 , 2012                                TARGT
```

```
2012 IK(1) = I + 1                                           TARGT
2013 IK(2) = J + 1                                           TARGT
2014 IK(3) = K + 1                                           TARGT
2015 DO    2020   L = 1 , 3                                  TARGT
2016 N = IK(L)                                               TARGT
2017 XX(L) = XMAN(N)                                         TARGT
2018 YY(L) = YMAN(N)                                         TARGT
2019 XS(L) = XX(L) + RANDOM(1)                               TARGT
2020 YS(L) = YY(L) + RANDOM(1)                               TARGT
2021 R3 = ( XS(1) + XS(2) + XS(3) )/3.0                      TARGT
2022 R4 = ( YS(1) + YS(2) + YS(3) )/3.0                      TARGT
2023 R5 = R3 + RANDOM(2)                                     TARGT
2024 R6 = R4 + RANDOM(3)                                     TARGT
2025 X  = (FLOATF( IMAX + 1 )*ZETA)/2.0                      TARGT
2026 DO    2030   L = 1 , IMAX                               TARGT
2027 XIMP(L)  = RANDOM(4) + R5 - X + ZETA*FLOATF(L)          TARGT
2028 YIMP(L)  = RANDOM(5) + R6                               TARGT
2029 XIMPW(L) = XIMP(L)*COST + YIMP(L)*SINT                  TARGT
2030 YIMPW(L) = YIMP(L)*COST - XIMP(L)*SINT                  TARGT
2031 R1 = ( XX(1) + XX(2) + XX(3) )/3.0                      TARGT
2032 R2 = ( YY(1) + YY(2) + YY(3) )/3.0                      TARGT
2033 GO  TO  52                                              TARGT
```

```
C3001            COMPUTATION  OF  FRAGMENTARY  EFFECTS                FRAGM
 3002 DO    3029    N = 1 , NMAX                                      FRAGM
 3003 DO    3029    I = 1 , IMAX                                      FRAGM
 3004 IF    ( YIMP(I) - YMAN(N) )   3005 , 3005 , 3029                FRAGM
 3005 X  =  XMAN(N)  -  XIMP(I)                                       FRAGM
 3006 Y  =  YMAN(N)  -  YIMP(I)                                       FRAGM
 3007 DSQR = X*X + Y*Y                                                FRAGM
 3008 IF    ( DSQR - RSM )          3009 , 3009 , 3029                FRAGM
 3009 IF    ( DSQR - DELTA )        3021 , 3021 , 3010                FRAGM
 3010 IF  (Y)                       3013 , 3011 , 3013                FRAGM
 3011 BETA  =  9.9E35                                                 FRAGM
 3012 GO  TO  3014                                                    FRAGM
 3013 BETA  =  ABSF( X/Y )                                            FRAGM
 3014 ITAU  =  1                                                      FRAGM
 3015 IXI  =  1                                                       FRAGM
 3016 IF    ( ALPHA(ITAU,1) - BETA )     3017 , 3017 , 3029           FRAGM
 3017 IF    ( ALPHA(ITAU,2) - BETA )     3026 , 3018 , 3018           FRAGM
 3018 IF    ( RSQR(ITAU,IXI) - DSQR )    3023 , 3019 , 3019           FRAGM
 3019 PROB = RANDOM(13)                                               FRAGM
 3020 IF    ( DOS(ITAU,IXI) - PROB )     3029 , 3029 , 3021           FRAGM
 3021 D(4,N) = D(4,N) + 1.0                                           FRAGM
 3022 GO  TO  3029                                                    FRAGM
```

```
 3023 IF    ( IXI - IXIMX(ITAU) )          3024 , 3029 , 3029      FRAGM
 3024 IXI  =  IXI  +  1                                            FRAGM
 3025 GO  TO  3018                                                 FRAGM
 3026 IF    ( ITAU - ITMAX )               3027 , 3029 , 3029      FRAGM
 3027 ITAU = ITAU + 1                                              FRAGM
 3028 GO  TO  3015                                                 FRAGM
 3029 CONTINUE                                                     FRAGM
 3030 GO  TO  54                                                   FRAGM
C4001                  COMPUTATION  OF  SPLASH  EFFECTS            SPLSH
 4002 DO    4028   N = 1 , NMAX                                    SPLSH
 4003 DO    4028   I = 1 , IMAX                                    SPLSH
 4004 K = KMAX                                                     SPLSH
 4005 R0 = YMAN(N) - YIMP(I)                                       SPLSH
 4006 R1 = R0 - AMBDA(K)                                           SPLSH
 4007 R2 = XMAN(N) - XIMP(I)                                       SPLSH
 4008 IF    ( ABSF(R1) - A(K) )  4009 , 4009 , 4028                SPLSH
 4009 IF    ( ABSF(R2) - B(K) )  4010 , 4010 , 4028                SPLSH
 4010 R2 = R2*R2                                                   SPLSH
 4011 R5 = R1 + C(K)                                               SPLSH
 4012 R6 = R1 - C(K)                                               SPLSH
 4013 R3 = R2 + R5*R5                                              SPLSH
 4014 R4 = R2 + R6*R6                                              SPLSH
```

```
4015 RHO = SQRTF(R3) + SQRTF(R4)                                SPLSH
4016 IF   ( RHO - 2.0A(K) )    4017 , 4017 , 4028               SPLSH
4017 IF   ( K - 1 )            4018 , 4027 , 4018               SPLSH
4018 K = K - 1                                                  SPLSH
4019 R1 = R0 - AMBDA(K)                                         SPLSH
4020 R5 = R1 + C(K)                                             SPLSH
4021 R6 = R1 - C(K)                                             SPLSH
4022 R3 = R2 + R5*R5                                            SPLSH
4023 R4 = R2 + R6*R6                                            SPLSH
4024 RHO = SQRTF(R3) + SQRTF(R4)                                SPLSH
4025 IF   ( RHO - 2.0A(K) )    4017 , 4017 , 4026               SPLSH
4026 K = K + 1                                                  SPLSH
4027 D(1,N) = D(1,N) + S(K)                                     SPLSH
4028 CONTINUE                                                   SPLSH
4029 GO  TO  56                                                 SPLSH
C5001      COMPUTATION  OF  TIME  DEPENDENT  VAPOR  EFFECTS     VAPOR
5002 DO    5087   N = 1 , NMAX                                  VAPOR
5003 BPS = 1.0                                                  VAPOR
5004 DO    5087   I = 1 , IMAX                                  VAPOR
5005 X = XMANW(N) - XIMPW(I)                                    VAPOR
5006 Y = YMANW(N) - YIMPW(I)                                    VAPOR
5007 IF   ( ABSF(Y) - 2.0*R )  5008 , 5008 , 5087               VAPOR
```

```
 5008 IF   ( X + R )           5087 , 5087 , 5009                VAPOR
 5009 IF   ( X - 1000.0 )      5010 , 5087 , 5087                VAPOR
 5010 IF  (BPS)                5047 , 5047 , 5012                VAPOR
C5011             COMPUTATION  OF  BREATHING  PARAMETERS          BPARA
 5012 BPS = 0.0                                                  BPARA
 5013 TW = DELT*INTF((T1P*RANDOM(13) + DELT2)/DELT)              BPARA
 5014 IF   ( TW - T1P )        5016 , 5015 , 5016                BPARA
 5015 TW = 0.0                                                   BPARA
 5016 TM1 = DELT*INTF((RANDOM(6) + DELT2)/DELT)                  BPARA
 5017 TM1P = TM1 + TW                                            BPARA
 5018 TM = DELT*INTF((RANDOM(7) + DELT2)/DELT) + TM1             BPARA
 5019 TMP = TM + TW                                              BPARA
 5020 IF   ( TM1P - TPP )      5021 , 5021 , 5023                BPARA
 5021 TH1P = TPP                                                 BPARA
 5022 GO  TO  5036                                               BPARA
 5023 IF   ( 2.0*TM1P - TPP - T1P )     5024 , 5024 , 5026       BPARA
 5024 TH1P = TM1P                                                BPARA
 5025 GO  TO  5036                                               BPARA
 5026 IF   ( TM1P - T1P )               5027 , 5027 , 5029       BPARA
 5027 TH1P = T1P + TPP/2.0                                       BPARA
 5028 GO  TO  5036                                               BPARA
 5029 IF   ( TM1P - T1P - TPP )         5030 , 5030 , 5032       BPARA
```

```
 5030 TH1P = T1P + TPP                                                    BPARA
 5031 GO  TO  5036                                                        BPARA
 5032 IF   ( 2.0*TM1P - 3.0*T1P - TPP ) 5033 , 5033 , 5035                BPARA
 5033 TH1P = TM1P                                                         BPARA
 5034 GO  TO  5036                                                        BPARA
 5035 TH1P = 2,0T1P + TPP/2.0                                             BPARA
 5036 J = 1                                                               BPARA
 5037 BHT = DELT*INTF((RANDOM(8) + DELT2)/DELT)                           BPARA
 5038 TGP(J) = TH1P + BHT                                                 BPARA
 5039 GO  TO  5043                                                        BPARA
 5040 J = J + 1                                                           BPARA
 5041 BHT = DELT*INTF((RANDOM(8) + DELT2)/DELT)                           BPARA
 5042 TGP(J) = TGP(J-1) + GD + BHT                                        BPARA
 5043 IF   ( TGP(J) - TMP )              5040 , 5044 , 5044               BPARA
 5044 TGP(J) = 9.9E35                                                     BPARA
 5045 TV = RANDOM(9)                                                      BPARA
 5046 IF   ( X - R )                     5047 , 5047 , 5049               VAPOR
 5047 TA = 0.0                                                            VAPOR
 5048 GO  TO  5050                                                        VAPOR
 5049 TA = DELT*INTF((( X - R )/UBAR + DELT2)/DELT)                       VAPOR
 5050 TD = DELT*INTF((( X + R )/UBAR + DELT2)/DELT)                       VAPOR
C5051      COMPUTATION  OF  TIME  DEPENDENT  VAPOR  DOSAGE                DOSE
```

```
5052 TEP = TA + TW                                              DOSE
5053 IF   ( TEP - TMP )                  5054 , 5087 , 5087     DOSE
5054 IF   ( TEP - TH1P )                 5055 , 5073 , 5073     DOSE
5055 MGMAX = ( TH1P - TEP )/DELT                                DOSE
5056 K = 1                                                      DOSE
5057 T = TA + DELT2                                             DOSE
5058 MQ = XMODF(XFIXF(TEP/DELT),MQMAX) + 1                      DOSE
5059 IF   ( DF(MQ) )                     5064 , 5064 , 5060     DOSE
5060 IF   ( T - TD )                     5061 , 5087 , 5087     DOSE
5061 L = 1                                                      DOSE
5062 GO  TO  6002                                               DOSE
5063 D(2,N) = D(2,N) + DF(MQ)*CHI*TV                            DOSE
5064 IF   ( K - MGMAX )                  5065 , 5073 , 5065     DOSE
5065 K = K + 1                                                  DOSE
5066 T = T + DELT                                               DOSE
5067 IF   ( T - TM )                     5068 , 5068 , 5087     DOSE
5068 IF   ( MQ - MQMAX )                 5071 , 5069 , 5071     DOSE
5069 MQ = 1                                                     DOSE
5070 GO  TO  5059                                               DOSE
5071 MQ = MQ + 1                                                DOSE
5072 GO  TO  5059                                               DOSE
5073 J = 0                                                      DOSE
```

```
5074 J = J + 1                                                  DOSE
5075 IF   ( TGP(J) - 9.9E35 )           5076 , 5087 , 5076      DOSE
5076 T = TGP(J) - TW                                            DOSE
5077 IF   ( T + GD - TA )               5074 , 5074 , 5078      DOSE
5078 T = T - DELT2                                              DOSE
5079 DO    5085   K = 1 , MKMAX                                 DOSE
5080 T = T + DELT                                               DOSE
5081 IF   ( T - TD )                    5082 , 5087 , 5087      DOSE
5082 IF   ( T - TM )                    5083 , 5083 , 5087      DOSE
5083 L = 2                                                      DOSE
5084 GO  TO  6002                                               DOSE
5085 D(3,N) = D(3,N) + DVG(K)*CHI                               DOSE
5086 GO  TO  5074                                               DOSE
5087 CONTINUE                                                   VAPOR
5088 GO  TO  58                                                 VAPOR
C6001        COMPUTATION  OF  THE  CONCENTRATION  OF  THE  GAS  CHIF
6002 IF   ( T - 120.0 )                 6003 , 5087 , 5087      CHIF
6003 R1 = UBAR*T                                                CHIF
6004 R3 = ( X - R1 )*( X - R1 ) + Y*Y                           CHIF
6005 R2 = ( ( R1 + DR )*0.01 )**ALPA                            CHIF
6006 R4 = R2*R2                                                 CHIF
6007 CHI = ( Q*EXPF( -R3/(23.2562*R4) ) )/( 402.90378*R4*R2 )   CHIF
```

```
6008 GO  TO  ( 5063 , 5085 ) , L                                     CHIF
     END
*    FORTRAN
C8001                      RANDOM  NUMBER  GENERATOR                 RN
8002 SUBROUTINE  RN( K , M , N , X )                                 RN
8003 K = 8193*K + 1                                                  RN
8004 X = FLOATF(K)/131072.0                                          RM
8005 M = 40.0*X                                                      RN
8006 Y = INTF(1000.0*X)                                              RM
8007 N = Y - 25.0*FLOATF(M)                                          RN
8008 RETURN                                                          RN
     END
*    FORTRAN
C1001  SELECTION  OF  RANDCM  VALUES  FROM  PROBABILITY  DISTRIBUTIONS  RANDM
1002 FUNCTION  RANDOM(I)                                             RANDM
1003 DIMENSION  A(12,42)                                             RANDM
1004 CALL  RN( K , M , N , X )                                       RANDM
1005 IF   ( I - 15 )      1009 , 1029 , 1006                         RANDM
1006 K = I                                                           RANDM
1007 J = I                                                           RANDM
1008 GO  TO  1035                                                    RANDM
1009 IF  (I)              1035 , 1010 , 1013                         RANDM
```

```
1010 READ  INPUT  TAPE IN , 1037 , RAN1 , RAN2 , RAN3                RANDM
1011 READ  INPUT  TAPE IN , 1038 , ((A(M,N),N=1,42),M=1,12)          RANDM
1012 GO  TO  1035                                                    RANDM
1013 IF   ( I - 13 )      1014 , 1019 , 1024                         RANDM
1014 IF  (RAN3)           1015 , 1015 , 1017                         RANDM
1015 RA = A(I,42)                                                    RANDM
1016 GO  TO  1035                                                    RANDM
1017 RA = A(I,M+1) + 0.04*FLOATF(N)*(A(I,M+2) - A(I,M+1))            RANDM
1018 GO  TO  1035                                                    RANDM
1019 IF  (RAN3)           1020 , 1020 , 1022                         RANDM
1020 RA = RAN1                                                       RANDM
1021 GO  TO  1035                                                    RANDM
1022 RA = X                                                          RANDM
1023 GO  TO  1035                                                    RANDM
1024 IF  (RAN3)           1025 , 1025 , 1027                         RANDM
1025 RA = RAN2                                                       RANDM
1026 GO  TO  1035                                                    RANDM
1027 RA = 1000.0*X                                                   RANDM
1028 GO  TO  1035                                                    RANDM
1029 WRITE  OUTPUT  TAPE  IO , 1039                                  RANDM
1030 DO    1032   N = 1 , 42                                         RANDM
1031 V = 0.025*FLOATF(N-1)                                           RANDM
```

```
1032 WRITE  OUTPUT  TAPE  IO , 1040 , V , (A,M,N),M=1,12)        RANDM
1033 WRITE  OUTPUT  TAPE  IO , 1041 , J , K                       RANDM
1034 GO  TO  1036                                                 RANDM
1035 RANDOM = RA                                                  RANDM
     WRITE  OUTPUT  TAPE  IO , 1060 , I , K , M , N , RA
1060 FORMAT(1H ,4I10,F20.7)
1036 RETURN                                                       RANDM
1037 FORMAT(3F20.7)                                               RANDM
1038 FORMAT(F11.3,6F10.3)                                         RANDM
1039 FORMAT(1H1////46(1H ),27HPROBABILITY  DISTRIBUTIONS          RANDM
   1       40H0  X          1          2          3               RANDM
   2       40H   4          5          6          7               RANDM
   3       40H 8        9        10        11        12   /)      RANDM
1040 FORMAT(1H ,F5.3,5F10.3,7F9.3)                                RANDM
1041 FORMAT(1H0//30(1H ),28HINITIAL  RANDOM  NUMBER  WAS,I8,      RANDM
   1       35H    LAST  RANDOM  NUMBER  USED  WAS,I8)             RANDM
     END
)1 *XP  - -X*GP                                                   GLTAN201
*J ,*(*,(*N 9-X7 97 -7  9-X7 74 =7 -4 -4-X5497-XD-X7 77 -6 7E-(4 -6-(7 -GLTAN202
*J7)7(*7GGG E-XG-(76-4-X7 97 -4 7-777-X7-X9-(46-9-(=-(7-(E-(4-(66-9-(-6-GLTAN203
*J7PP(*-    4-X--(E-(4-(G-(549**X())P    =(7*-  **(P- P- *+ *+ *+ P+ *  GLTAN204
*6776-      *  *  P                                               GLTAN205
```

* DATA

0.55555555 333.33333 1.0

-129.070	-93.024	-78.194	-68.320	-60.813	-54.594	-49.224	*	101
-44.357	-39.959	-35.851	-32.020	-28.371	-24.879	-21.540	*	102
-18.288	-15.144	-12.019	-8.951	-5.954	-2.972	0.000	*	103
2.972	5.954	8.951	12.019	15.144	18.288	21.540	*	104
24.879	28.371	32.020	35.851	39.959	44.357	49.224	*	105
54.594	60.813	68.320	78.194	93.024	129.070	40.000	*	106
-149.232	-107.554	-90.407	-78.990	-70.310	-63.120	-56.911	*	201
-51.282	-46.198	-41.447	-37.018	-32.799	-28.761	-24.901	*	202
-21.140	-17.505	-13.892	-10.344	-6.879	-3.432	0.000	*	203
3.432	6.879	10.345	13.892	17.505	21.140	24.901	*	204
28.761	32.799	37.018	41.447	46.198	51.282	56.911	*	205
63.120	70.310	78.990	90.407	107.554	149.232	-30.000	*	206
-274.002	-197.640	-166.163	-145.134	-129.228	-116.013	-104.493	*	301
-94.248	-84.911	-76.185	-68.032	-60.288	-52.861	-45.773	*	302
-38.855	-32.173	-25.539	-19.008	-12.653	-6.323	0.000	*	303
6.323	12.653	19.008	25.539	32.173	38.855	45.773	*	304
52.861	60.288	68.032	76.185	84.911	94.248	104.493	*	305
116.013	129.228	145.134	166.163	197.640	274.002	50.000	*	306
-57.452	-40.706	-34.212	-29.896	-26.611	-23.886	-21.520	*	401
-19.412	-17.485	-15.691	-14.014	-12.417	-10.889	-9.426	*	402

-8.003	-6.624	-5.260	-3.919	-2.606	-1.302	0.000	*	403
1.302	2.606	3.919	5.260	6.624	8.003	0.426	*	404
10.889	12.417	14.014	15.691	17.485	19.412	21.520	*	405
23.886	26.611	29.896	34.212	40.706	57.452	-15.000	*	406
-112.936	-81.396	-68.420	-59.780	-53.212	-47.770	-43.071	*	501
-38.812	-34.964	-31.369	-28.017	-24.825	-21.769	-18.848	*	502
-16.002	-13.251	-10.516	- 7.832	-5.210	-2.601	0.000	*	503
2.601	5.210	7.832	10.516	13.251	16.002	18.848	*	504
21.769	24.825	28.017	31.369	34.964	38.812	43.071	*	505
47.770	53.212	59.780	68.420	81.396	112.936	30.000	*	506
0.100	0.131	0.163	0.194	0.207	0.216	0.225	*	601
0.234	0.243	0.252	0.261	0.270	0.279	0.288	*	602
0.296	0.305	0.313	0.321	0.329	0.337	0.345	*	603
0.353	0.361	0.369	0.377	0.385	0.394	0.402	*	604
0.410	0.418	0.427	0.435	0.443	0.452	0.460	*	605
0.468	0.477	0.485	0.493	0.517	0.600	0.300	*	606
8.500	9.188	9.500	9.708	9.917	10.150	10.400	*	701
10.625	10.833	11.050	11.300	11.500	11.800	12.050	*	702
12.300	12.550	12.800	13.042	13.250	13.458	13.700	*	703
13.950	14.200	14.450	14.700	14.950	15.200	15.450	*	704
15.750	16.063	16.375	16.875	17.250	17.625	18.500	*	705
19.250	20.500	21.750	22.500	23.750	25.000	15.000	*	706

2.000	3.125	3.750	4.250	4.750	5.375	6.000	*	801
6.505	7.000	7.583	8.000	8.417	8.833	9.250	*	802
9.666	10.083	10.500	10.917	11.333	11.875	12.500	*	803
13.125	13.750	14.188	14.500	15.125	15.750	16.375	*	804
17.000	17.583	18.000	18.750	19.500	20.125	20.750	*	805
21.375	22.000	22.875	24 000	25.250	26.500	12.000	*	806
0.600	0.636	0.671	0.707	0.743	0.779	0.813	*	901
0.844	0.875	0.905	0.930	0.955	0.980	1.004	*	902
1.023	1.042	1.062	1.081	1.100	1.119	1.138	*	903
1.158	1.177	1.196	1.217	1.238	1.258	1.279	*	904
1.300	1.328	1.356	1.383	1.414	1.450	1.486	*	905
1.538	1.600	1.663	1.750	1.950	2.300	1.500	*	906
0.000	0.000	0.000	0.000	0.000	0.000	0.000	*	1001
0.000	0.000	0.000	0.000	0.000	0.000	0.000	*	1002
0.000	0.000	0.000	0.000	0.000	0.000	0.000	*	1003
0.000	0.000	0.000	0.000	0.000	0.000	0.000	*	1004
0.000	0.000	0.000	0.000	0.000	0.000	0.000	*	1005
0.000	0.000	0.000	0.000	0.000	0.000	0.000	*	1006
0.000	0.000	0.000	0.000	0.000	0.000	0.000	*	1101
0.000	0.000	0.000	0.000	0.000	0.000	0.000	*	1102
0.000	0.000	0.000	0.000	0.000	0.000	0.000	*	1103
0.000	0.000	0.000	0.000	0.000	0.000	0.000	*	1104

0.000	0.000	0.000	0.000	0.000	0.000	0.000	* 1105
0.000	0.000	0.000	0.000	0.000	0.000	0.000	* 1106
0.000	0.000	0.000	0.000	0.000	0.000	0.000	* 1201
0.000	0.000	0.000	0.000	0.000	0.000	0.000	* 1202
0.000	0.000	0.000	0.000	0.000	0.000	0.000	* 1203
0.000	0.000	0.000	0.000	0.000	0.000	0.000	* 1204
0.000	0.000	0.000	0.000	0.000	0.000	0.000	* 1205
0.000	0.000	0.000	0.000	0.000	0.000	0.000	* 1206
1225.0	0.25	1.6	2.9	0.1			
11.5	2.0	0.88	45.0	1.4			
20.0	1.0	1.0	1.0	1.0			
3	6	3	12	10	5671		
6.0	14.0	10.0	9.8	2.0			
6.0	16.0	12.0	10.6	1.0			
6.0	18.0	14.0	11.3	0.5			
0.0	33.0						
0.0	67.0						
33.0	0.0						
33.3	33.0						
33.3	67.0						

```
     33.0      100.0
     67.0        0.0
     67.0       33.0
     67.0       67.0
     67.0      100.0
    100.0       33.0
    100.0       67.0
   0.01    0.02    0.03    0.04    0.07    0.09    0.12    0.12    0.12    0.12
   0.09    0.07    0.04    0.03    0.02    0.01    0.00    0.00    0.00    0.00
   0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00
   0.01    0.02    0.03    0.04    0.07    0.09    0.12    0.12    0.12    0.12
   0.09    0.07    0.04    0.03    0.02    0.01    0.00    0.00    0.00    0.00
   0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00
5 90.0 80.0  1.0 0.95  4.00 0.70   9.0 0.30  25.0 0.075  110.25 0.025   *   13
5 60.0 30.0 25.0 0.95 90.25 0.70 400.0 0.30 625.0 0.075 1225.00 0.025   *   14
5 10.0  5.0 16.0 0.95 49.00 0.70 225.0 0.30 400.0 0.075 1024.00 0.025   *   15
                                                                           432
```